Core Information Model (CoreModel)

TR-512.6

Physical

Version 1.6
January 2024

ONF Document Type: Technical Recommendation
ONF Document Name: Core Information Model version 1.6

Disclaimer

Open Networking Foundation
1000 El Camino Real, Suite 100, Menlo Park, CA 94025
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

Disclaimer **2**

Important note **2**

Document History **6**

1 Introduction to the document suite **7**

1.1 References 7
1.2 Definitions 7
1.3 Conventions 7
1.4 Viewing UML diagrams 7
1.5 Understanding the figures 7

2 Introduction to the Physical model **8**

3 Physical model detail **8**

3.1 Equipment Pattern 8
3.1.1 Equipment 11
3.1.2 Holder 12
3.1.3 Connector 12
3.1.4 Cable 13
3.2 Equipment Detail 14
3.2.1 Invariant Equipment Detail 15
3.2.1.1 EnvironmentalRating 15
3.2.1.2 EquipmentInstance 16
3.2.1.3 EquipmentStructure 16
3.2.1.4 EquipmentType 17
3.2.1.5 HolderStructure 17
3.2.1.6 ManufacturedThing 18
3.2.1.7 ManufacturerProperties 18
3.2.1.8 MechanicalFeatures 19
3.2.1.9 OperatorAugmentedEquipmentInstance 19
3.2.1.10 OperatorAugmentedEquipmentType 19
3.2.1.11 PhysicalCharacteristics 20
3.2.1.12 Position 20
3.2.1.13 SpatialPropertiesOfType 20
3.2.1.14 Swappability 21
3.2.2 Dynamic Equipment Detail 21
3.2.2.1 FunctionEnablers 21

3.2.2.2 HolderMonitor 22
3.2.2.3 Location 22
3.2.2.4 MechanicalFunctions 22
3.2.2.5 PhysicalProperties 23
3.3 Connector to LTP Model 23
3.3.1 The AccessPort 23
3.3.1.1 Changes since 1.3.1 24
3.3.2 The MultipleStrandSpan 24
3.3.3 AccessPort and MultipleStrandSpan 26
3.3.4 Detailed description of the model fragment 27
3.3.4.1 AccessPort 27
3.3.4.2 ElementalSignals 27
3.3.4.3 GroupOfPins 27
3.3.4.4 GroupOfStrands 27
3.3.4.5 MultipleStrandSpan 27
3.3.4.6 Pin 28
3.3.4.7 PinGroup 28
3.3.4.8 PinLayout 28
3.3.4.9 SignalRefPt 28
3.3.4.10 SignalRefPtGroup 28
3.3.4.11 Strand 28
3.4 Equipment to Function Sketch 29
3.4.1 Equipment to Function using ProcessingConstruct and ConstraintDomain 29
3.4.2 "Equipment Protection" 30
3.4.3 Classes associated with Equipment to function mapping with resilience 31
3.4.3.1 ProcessingConstruct (PC) 31
3.4.3.2 PcPort 32
3.4.3.3 ForwardingConstruct (FC) 33
3.4.3.4 FcSwitch 33
3.4.3.5 Equipment 34
3.4.4 V1.4 refinements of V1.3 model 35
3.4.5 Obsolete model from V1.2 use to illustrate the V1.3 model 35
3.4.5.1 AggregateFunction 36
3.4.5.2 AtomicFunction 36
3.4.5.3 FunctionBlock 36
3.4.5.4 ResilienceSelector 37
3.4.5.5 ResilientFunctionBlock 37
3.5 FRU and non-FRU 37
3.5.1 FieldReplaceable 38

3.5.2 NonFieldReplaceable 38
3.6 Connector Rules 39
3.6.1 ConnectorCableEnd 40
3.6.2 ConnectorInHolder 40
3.6.3 ConnectorOnEquipmentForCable 41
3.6.4 ConnectorOnEquipmentForHolder 41
3.7 Expected and Actual 41
3.7.1 ActualEquipment 42
3.7.2 ActualHolder 42
3.7.3 ExpectedEquipment 43
3.7.4 ExpectedHolder 43
3.8 Specification 43
3.8.1 NonFruSupportPosition 44
3.8.2 SupportConstraints 44
3.8.3 SupportedEquipmentType 44
3.8.4 SupportedNonFruType 45
3.9 Physical Connector and conceptual Port 45

4 Work in progress (see also TR-512.FE) 48
4.1 Addressing 48
4.2 Physical to functional model 48
4.3 Actual v expected 50

List of Figures

Figure 3-1 Skeleton Class Diagram of key object classes 9
Figure 3-2 - Equipment and Holder example 10
Figure 3-3 – Inside the Equipment 11
Figure 3-4 Equipment Detail Structure 15
Figure 3-5 Connector to LTP 24
Figure 3-6 Cable to FC 26
Figure 3-7 Connector/Cable to LTP/FC 26
Figure 3-8 Equipment to Function using ProcessingConstruct and showing Resilience 29
Figure 3-9 "Equipment Protection" 31
Figure 3-10 Equipment to Function 36
Figure 3-11 FRU and Non-FRU rules 38
Figure 3-12 Connector rules 40
Figure 3-13 Expected and actual 42
Figure 3-14 Specification 44
Figure 3-15 Basic cases of Access Port Reference 46

Figure 3-16 More Complex cases of intertwined Connectors 47
Figure 3-17 Unidirectional Cases 47
Figure 4-1 Connector/Port based addressing of LTPs 48
Figure 4-2 Simplified sketch of physical to functional 49
Figure 4-3 Simplified sketch of forms of resilience in an NE 50
Figure 4-4 Expectation v actual showing mismatch and blocking 51

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.
1.4	November 2018	Enhancements to connector model
1.5	September 2021	Enhancements to model structure
1.6	January 2024	Updated release and dates.

1 Introduction to the document suite

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 as well as 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 Physical model

The focus of this document is the modeling of physical things, especially equipment, in the ONF-CIM.

Note:

- A majority of the Physical model is experimental at this stage (some key classes and attributes have been upgraded to Mature/Preliminary since V1.2). It was considered vital to publish the work in progress on equipment as it is clearly an important part of the overall model. Many of the attributes and classes are not fully documented in the model.
- The Physical model deals with physical things where a physical thing is something that can be "measured with a ruler"[1]

This document:

- Introduces the Physical model structure
- Describes the key classes of the Physical model
- Explains the attributes of the Physical model
- Describes the relationship between the Connector and the LogicalTerminationPoint (LTP)
- Shows how the model deals with the relationship between physical and functional views
- Explains how the specification model describes equipment schemes (rules etc)
- Highlights work in progress to further advance the Physical model

The Physical model relates to:

- The Core Network Model including Termination and Forwarding described in TR-512.2 and Topology described in TR-512.4.
- The generalized processing and constraint model described in TR-512.11
- The specification model described in TR-512.7.

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

3 Physical model detail

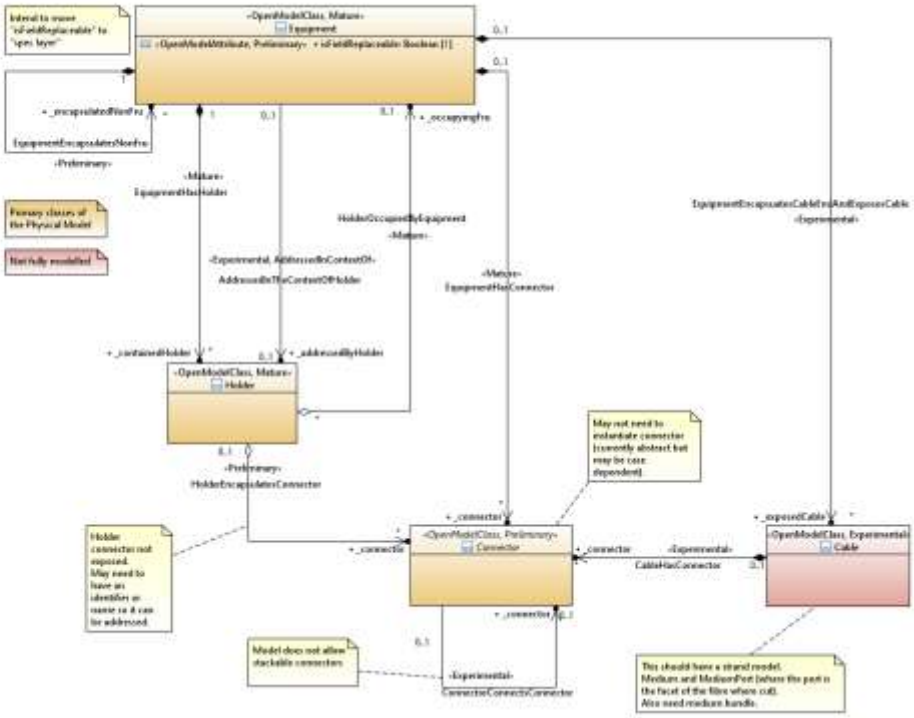
This section starts with a basic view of the equipment classes, then progresses through detail to sophisticated (and highly experimental) representations of equipment model constraints.

3.1 Equipment Pattern

The figure below sets out the basic equipment pattern.

[1] Often the word “physical” is used in the context of non-mechanical functional things. All non-mechanical functional things are considered under the functional model here regardless of how closely they are bound to the physical entities. All non-mechanical functional representations are considered as emergent abstractions and as virtualized (to some degree within a physical boundary). Mechanical functional things, such as fans, are not modelled in detail.

The classes of the model are described briefly after the figure. The associations are assumed to be sufficiently self-explanatory at this stage.



CoreModel diagram: Equipment-Pattern

Figure 3-1 Skeleton Class Diagram of key object classes

Taking a simple chassis, pictured in the figure below, as an example, we can consider the Holders to be the spaces within a piece of Equipment where a Holder is designed to accommodate another piece of Equipment. In a normal chassis it is possible that an Equipment may occupy several Holder, however it is not possible for a Holder to accommodate more than one Equipment at a time.

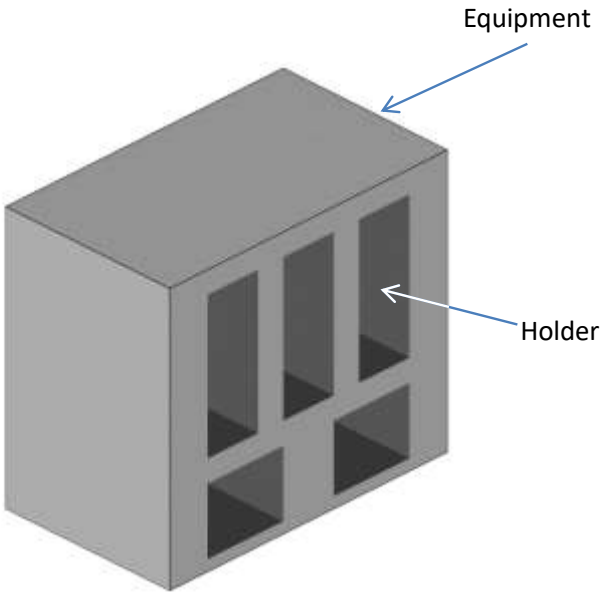


Figure 3-2 - Equipment and Holder example

There are two distinct roles for the Equipment entity controlled by the one attribute shown in the model figure above (isFieldReplaceable):

- Field Replaceable Unit (FRU):
  - Can be replaced in the field.
    - May be standalone
    - May plug in to a Holder in another Equipment (if not stand-alone)
- Non-Field Replaceable Unit (NFRU):
  - Cannot be replaced in the field. Is simply a subordinate part of an FRU (or another NFRU – where there must be an FRU at the top of the hierarchy).
  - Does not have any exposed Holders (any associated Holders are assumed to belong to the containing FRU).
  - Does not have any Connectors (any associated Connectors are assumed to belong to the containing FRU).

A method for representation of these restrictions is covered in section 3.5 FRU and non-FRU on page 37.

Connectors allow for Cables to be plugged in. So, for example, an SFP (Small Form-factor Pluggable) is a piece of Equipment that plugs in a Holder, and has a Connector on its front for a Cable with a Connector to be plugged in.

Looking at the picture below, we have removed some of the chassis panel to show the inside. We see that the circuit-pack (Equipment) will actually plug into Connectors on the (non-FRU) backplane. If we wish to explicitly represent the Connectors on the backplane, then we use association HolderEncapsulatesConnector to relate the Connector to the Holder it is at the back of.

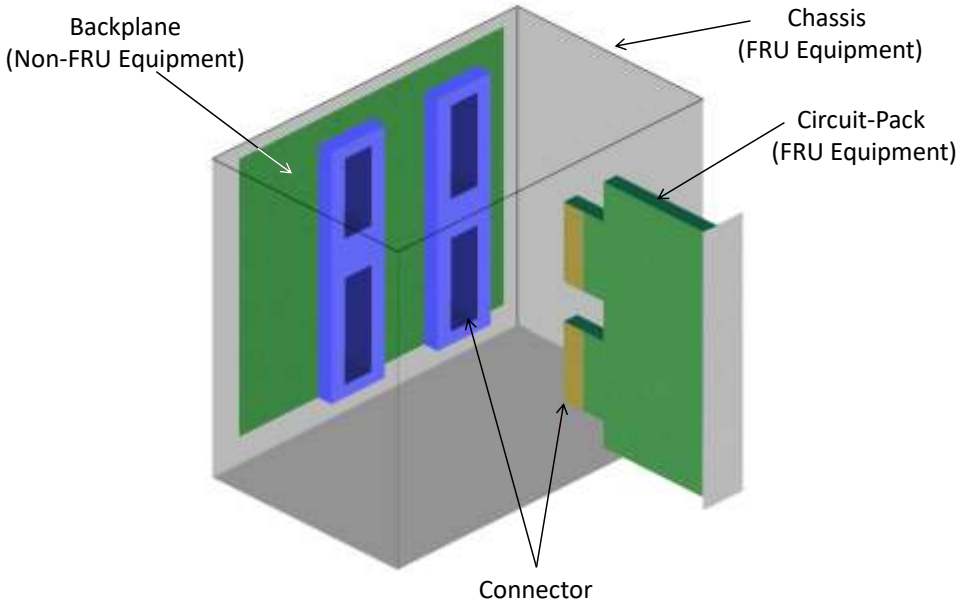


Figure 3-3 – Inside the Equipment

3.1.1 Equipment

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentPatternStructure::ObjectClasses::Equipment

Represents any relevant physical thing.
Can be either field replaceable or not field replaceable.
Note: The model is currently constrained to inside plant.

Inherits properties from:

- GlobalClass

This class is Mature.

Table 1: Attributes for Equipment

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_connector	Mature	A Connector on the Equipment.
_containedHolder	Mature	References the Holder in an Equipment that is available to take other Equipments. For example: - Slot in a sub-rack - Slot in a Field Replaceable Unit that can take a small form-factor pluggable.
_addressedByHolder	Experimental	A Holder through which the Equipment can be identified (where the Holder name/identifier is part of the Address).

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_encapsulatedNonFru	Preliminary	An Equipment that is part of this Equipment and that is not separately field replaceable (i.e. will be field replaced with this Equipment).
_exposedCable	Experimental	A Cable that is attached to part of the Equipment at one end and exposed at the other end through a Connector.
isFieldReplaceable	Preliminary	Indicates whether or not the equipment can be removed and replaced "in the field" (i.e. in a deployment) by normal operations personnel.

3.1.2 Holder

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentPatternStructure::ObjectClasses::Holder

Represents a space in an equipment in which another equipment can be fitted in the field.

Inherits properties from:

- LocalClass

This class is Mature.

Table 2: Attributes for Holder

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_connector	Preliminary	The connector associated with a Holder of an Equipment. May represent connector on a backplane that takes Field Replaceable Units or a connector on a circuit pack that takes an SFP (Small Form-factor Pluggable).
_occupyingFru	Mature	The FRU that is occupying the holder. A holder may be unoccupied. An FRU may occupy more hat one holder (using or blocking are intentionally not distinguished here).

3.1.3 Connector

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentPatternStructure::ObjectClasses::Connector

Represents a connector that may be fully exposed (e.g. to plug in a cable or on the end of a cable) or partially exposed (e.g. backplane to plug in another piece of equipment such as a module).
A physical port on the Equipment. A place where signals produced by the functionality of the Equipment may be accessed.
This class is abstract.

Inherits properties from:

- PinGroup
- LocalClass

This class is Preliminary.

Table 3: Attributes for Connector

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_connector	Preliminary	The Connector that is attached to this Connector so as to join the Equipment/Cable with this Connector to another Equipment/Cable. This may provide physical support and/or allow signal flow.
orientation	Experimental	The physical orientation of the connector, such as whether it is a male, or female, or neutral structure.
connectorType	Experimental	The type of the connector.
role	Experimental	The purpose of the Connector in the physical space and the functional space.

3.1.4 Cable

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentPatternStructure::ObjectClasses::Cable

Basic model representing a cable with connectors fitted at the cable ends as appropriate.
The cable may be an abstraction where the apparent ends are actually ends of two different cables that are connected indirectly by other cables and where that cable detail is not relevant or is not known.
This is intentionally a very basic representation of a cable.
The model does not account for any outside plant considerations.
In a more sophisticated representation cable ends might be represented that then associate to the attached connector.
At this point it is assumed that the basic model is sufficient.

Inherits properties from:

- GroupOfStrands

- GlobalClass

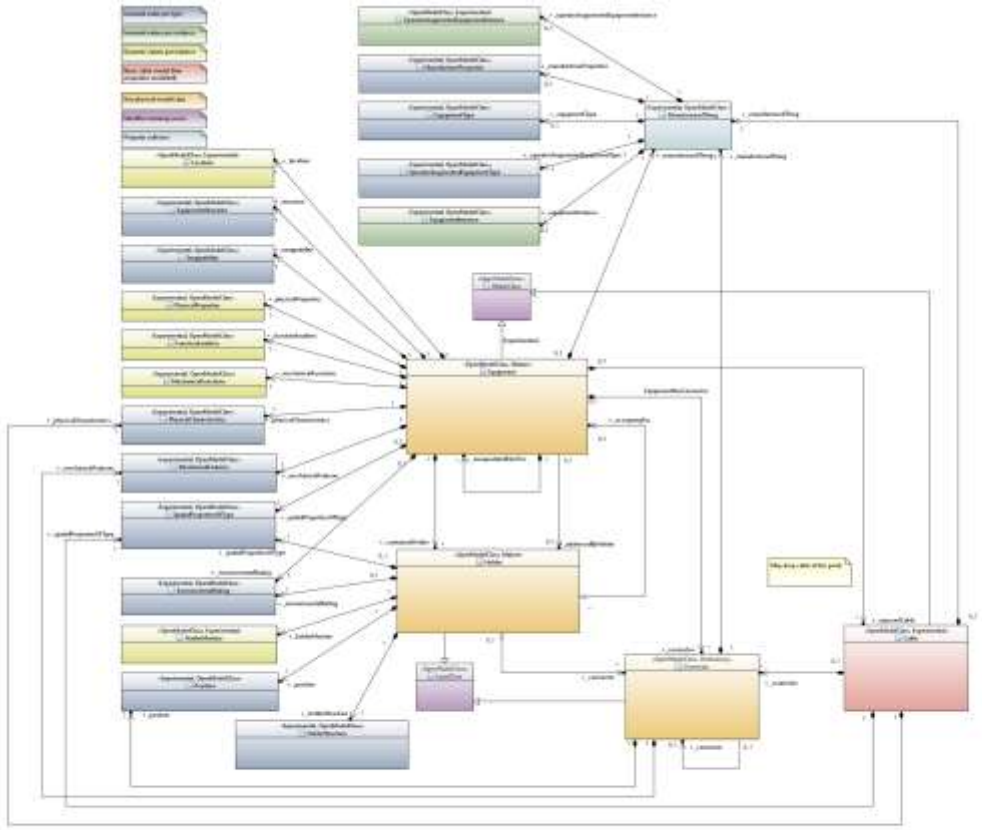
This class is Experimental.

Table 4: Attributes for Cable

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_connector	Preliminary	A connector that terminates the Cable to support the cable and/or allow signal flow into/out of the Cable.

3.2 Equipment Detail

The figure below highlights classes that group together related attributes (related as suggested by the name of the class). As noted in the key to the diagram the attributes are also grouped on the degree of variation. This latter grouping will guide the construction of specifications indicating what can reside only in the spec and what has to be available per instance (see TR-512.7 for more information)



CoreModel diagram: Equipment-DetailWithoutAttributes

Figure 3-4 Equipment Detail Structure

3.2.1 Invariant Equipment Detail

The following classes have attributes that do not change in value for the life of the Equipment. Some are the same value across all Equipment of the same type.

3.2.1.1 EnvironmentalRating

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::EnvironmentalRating

Represents the invariant physical operational boundaries for the equipment/holder type.
This class is Experimental.

Table 5: Attributes for EnvironmentalRating

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
thermalRating	Experimental	This attribute represents the thermal characteristics (preferred maximum/minimum, absolute maximum/minimum etc) that the entity can tolerate.
powerRating	Experimental	This attribute represents the power chracteristics (peak and mean per power source) of the entity. For an Equipment this is the power consumption. For a Holder this is the power that can be conveyed.
humidityRating	Experimental	This attribute represents the humidity characteristics (preferred maximum/minimum, absolute maximum/minimum etc.) that the entity can tolerate.

3.2.1.2 EquipmentInstance

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::EquipmentInstance

Represents the per instance invariant properties of the equipment.
This class is Experimental.

Table 6: Attributes for EquipmentInstance

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
manufactureDate	Experimental	This attribute represents the date on which this instance is manufactured.
serialNumber	Experimental	This attribute represents the serial number of this instance.
assetInstanceIdentifier	Experimental	This attribute represents the asset identifier of this instance from the manufacturer's perspective.

3.2.1.3 EquipmentStructure

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::EquipmentStructure

Represents the form of the equipment.
This class is Experimental.

Table 7: Attributes for EquipmentStructure

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
category	Experimental	This attribute provides the identifier for a category of equipments regarded as having particular shared characteristics.

3.2.1.4 EquipmentType

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::EquipmentType

Represents the invariant properties of the equipment that define and characterize the type.
This class is Experimental.

Table 8: Attributes for EquipmentType

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
description	Experimental	Text describing the type of Equipment.
modelIdentifier	Experimental	This attribute identifies the model of the equipment.
partTypeIdentifier	Experimental	This attribute identifies the part type of the equipment.
typeName	Experimental	This attribute identifies the type of the equipment.
version	Experimental	This attribute identifies the version of the equipment.

3.2.1.5 HolderStructure

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::HolderStructure

Represents the form of the holder.
This class is Experimental.

Table 9: Attributes for HolderStructure

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
holderCategory	Experimental	To be provided

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
isCaptive	Experimental	To be provided
isGuided	Experimental	This attribute indicates whether the holder has guides that constrain the position of the equipment in the holder or not.
isQuantisedSpace	Experimental	To be provided

3.2.1.6 ManufacturedThing

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::ManufacturedThing

Collects all invariant aspects of a manufactured thing.
This class is Experimental.

Table 10: Attributes for ManufacturedThing

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_manufacturerProperties	Experimental	See referenced class
_equipmentType	Experimental	See referenced class
_equipmentInstance	Experimental	See referenced class
_operatorAugmentedEquipmentType	Experimental	See referenced class
_operatorAugmentedEquipmentInstance	Experimental	See referenced class

3.2.1.7 ManufacturerProperties

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::ManufacturerProperties

Represents the properties of the manufacturer.
This class is Experimental.

Table 11: Attributes for ManufacturerProperties

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
manufacturerIdentifier	Experimental	To be provided
manufacturerName	Experimental	To be provided

3.2.1.8 MechanicalFeatures

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::MechanicalFeatures

Represents the invariant characteristics of dynamic mechanical features of a physical thing.
This class is Experimental.

3.2.1.9 OperatorAugmentedEquipmentInstance

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::OperatorAugmentedEquipmentInstance

Represents the invariant properties of the equipment asset allocated by the operator that define and characterize the type.
This class is Experimental.

Table 12: Attributes for OperatorAugmentedEquipmentInstance

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
assetInstanceIdentifier		This attribute represents the asset identifier of this instance from the operator's perspective.

3.2.1.10 OperatorAugmentedEquipmentType

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::OperatorAugmentedEquipmentType

Represents the invariant properties of the equipment asset allocated by the operator that define and characterize the type.
This class is Experimental.

Table 13: Attributes for OperatorAugmentedEquipmentType

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
assetTypeIdentifier	Experimental	To be provided

3.2.1.11 PhysicalCharacteristics

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::PhysicalCharacteristics

Represents the invariant physical characteristics (including composition and physical robustness) of the type.
This class is Experimental.

Table 14: Attributes for PhysicalCharacteristics

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
weightCharaceristics	Experimental	To be provided
fireCharacteristics	Experimental	To be provided
materials	Experimental	To be provided

3.2.1.12 Position

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::Position

Represents the invariant relative position of the holder (with respect to some frame of reference in an equipment) or connector on an equipment or pin in a connector.
This class is Experimental.

Table 15: Attributes for Position

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
relativePosition	Experimental	To be provided

3.2.1.13 SpatialPropertiesOfType

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::SpatialPropertiesOfType

Represents the basic invariant spatial properties of a physical thing.
This class is Experimental.

Table 16: Attributes for SpatialPropertiesOfType

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
height	Experimental	To be provided

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
width	Experimental	To be provided
length	Experimental	To be provided

3.2.1.14 Swappability

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::InvariantDetails::Swappability

Represents the degree of field replacement that is possible for the equipment type.
This class is Experimental.

Table 17: Attributes for Swappability

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
isHotSwappable	Experimental	To be provided

3.2.2 Dynamic Equipment Detail

The following classes have attributes that can change in value during the life of the Equipment.

3.2.2.1 FunctionEnablers

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::DynamicDetails::FunctionEnablers

Represents the dynamic aspects of the properties that relate to the motive force that directly enable functionality to emerge from the equipment.
This class is Experimental.

Table 18: Attributes for FunctionEnablers

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
powerState	Experimental	The state of the power being supplied to the equipment. Note that this attribute summarizes the power state. Full details on the actual power system would be provided from a number of PC instances representing the relevant parts of the Power function (e.g. different voltage supplies).

3.2.2.2 HolderMonitor

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::DynamicDetails::HolderMonitor

Represents the dynamic state of the holder instance.
This class is Experimental.

Table 19: Attributes for HolderMonitor

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
isActive	Experimental	Indicates that the holder is active and is supporting an Equipment instance.
isActualMismatchWithExpected	Experimental	Indicates that the equipment in the holder does not match the equipment expected to be in the holder.
_aggregateFunction	Obsolete Experimental	Obsolete
_supportingPc	Experimental	The functionality supporting this entity.

3.2.2.3 Location

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::DynamicDetails::Location

Represents where the equipment is.
This class is Experimental.

Table 20: Attributes for Location

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
equipmentLocation	Experimental	To be provided
geographicalLocation	Experimental	To be provided

3.2.2.4 MechanicalFunctions

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::DynamicDetails::MechanicalFunctions

Represents the dynamic aspects of the mechanical functions of the equipment.
This class is Experimental.

Table 21: Attributes for MechanicalFunctions

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
rotationSpeed	Experimental	To be provided

3.2.2.5 PhysicalProperties

Qualified Name: CoreModel::CorePhysicalModel::EquipmentDetail::ObjectClasses::DynamicDetails::PhysicalProperties

Represents the dynamic aspects of the physical environmental properties of the equipment. This class is Experimental.

Table 22: Attributes for PhysicalProperties

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
temperature	Experimental	To be provided

3.3 Connector to LTP Model

The figure in this section relates the physical aspects of the model at the boundary (connector, pin etc) to the functional aspects at the boundary (LTP, FC etc).

3.3.1 The AccessPort

The AccessPort is the most abstract level of the point-oriented aspect of the physical model and is a bridge between the point aspects of the functional model and the physical model.

Consider the figure below (where physical things are shown in orange and functional in green). As shown in the figure, the Connector, introduced earlier, is decomposed into pins, hence the connector is a PinGroup where the pins are owned by the connector (and are not meaningful by themselves). A pin is not meaningfully divisible. The pins are laid out in the connector and hence have position in the connector.

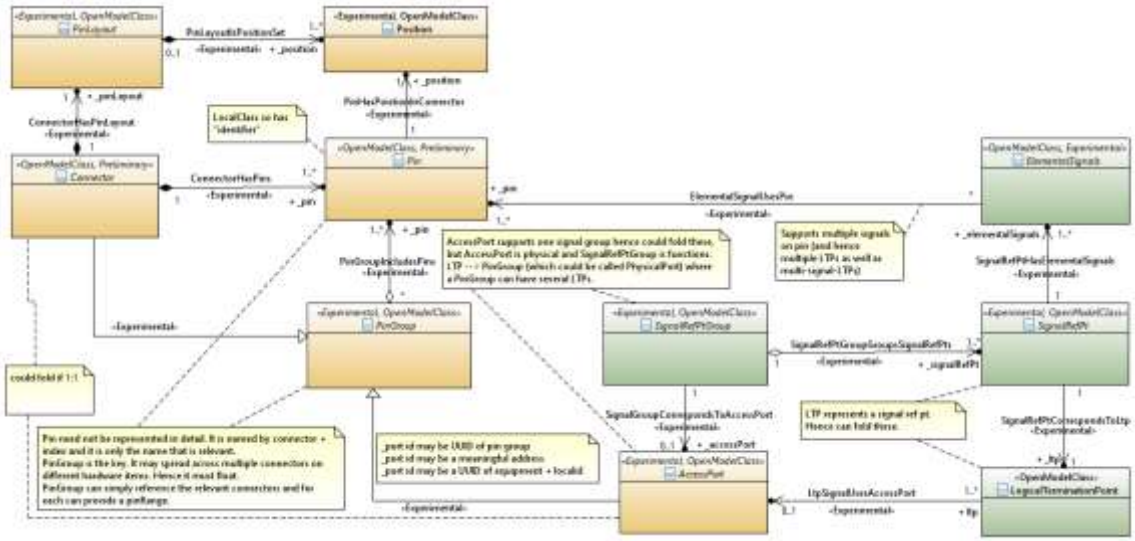
Examining the functional aspects of the figure, the LTP is a SignalReferencePoint which may carry 1 or more ElementalSignals. An ElementalSignal may require more than one Pin and a Pin may support more than one ElementalSignal. In a simple case, an LTP may process only one ElementalSignal and that Signal may be supported by a single Pin that is by itself in a Connector. In this simple case there is clearly a 1:1 relationship between the LTP and the Connector.

However, this is a corner case. In a majority of, cases, the LTP carries more than one ElementalSignal and the relationship between the LTP and the Pins of the Connectors is not

simple. Indeed, several signals may share, and be dependent on a single Pin or on several Pins. The groupings are such that if any pins were to be removed from the group, then all the signals of the group would be lost, and such that the signals supported by the PinGroup are indivisible when they flow through the pins.

To enable the necessary flexibility the Pins of the Connector are decoupled from the Signals of the LTP via the AccessPort. The AccessPort is a PinGroup where the Pins in the group may be scattered across several Connectors. The PinGroup is such that a group of LTPs depend upon "all" Pins in the group, i.e. if any pin is removed this would cause failure in signals such that the operation of all of the set of LTPs would be impacted (fail or degrade in some way).

The notes shown on the figure touch on some ongoing discussions.



CoreModel diagram: Equipment-ConnectorPinPortAndLTP

Figure 3-5 Connector to LTP

3.3.1.1 Changes since 1.3.1

The definitions of some of the classes have been corrected. As a result:

- The GroupOfPins class has been eliminated from the model (as PinGroup is the GroupOfPins)
- The AccessPort is a PinGroup (rather than the other way round – an error)
- The SignalRefPtGroup relates to the AccessPort rather than the PinGroup

3.3.2 The MultipleStrandSpan

The previous section dealt with the point aspect of the model. The relationship between the FC and the Cable is equivalent to the relationship between the LTP and the Connector.

The MultipleStrandSpan is the most abstract level of the arc-oriented aspect of the physical model and the bridge between the arc aspects of functional model and the physical model.

Consider the figure below (where physical things are shown in orange and functional in green). As shown in the figure, the Connector, introduced earlier, is decomposed into pins. The details of this aspect are covered in the previous section. The Pin is at the end of a Strand and the Connector at the end of a Cable.

It should be noted that a Cable may not have connectors at all/any ends and a Strand may not have Pins at all/any ends. The Cables and Strands of most interest in this model are the ones with Connectors and Pins respectively. Most of the other Cables/Strands are abstracted[2].

Where the Strand does not have Pins, it may be spliced (the model only supports simple splice cases), or simply aggregated into a group of adjacent Strands which another Strand abstracts. The assumption here is that the detail of the sequence of Strands is not relevant, but it may be relevant to know that the apparent Strand between two points is made of a sequence of Strands of different properties. Where it is relevant to understand the actual sequence the _splicedStrand attribute can be used to chain the Strands.

A Cable may have multiple strands and hence it is a GroupOfStrands where the Strands are owned by the Cable. A Strand is not meaningfully divisible. The Strand layout in the Cable is not considered relevant here, although it is expected that there may be some properties on a Cable that defines some aspect of Strand layout, e.g, twisted, coax. As a Cable can bundle Cables each internal segment of the Cable could be modelled if beneficial.

The elemental aspects of the FC are driven by the LTP. Clearly an FC can be decomposed into parts in the functional model. Here the relationship is simply from any FC to a GroupOfStrands that support the FC. The specific group is call the MultipleStrandSpan.

Examining the functional aspects of the figure in the previous section and considering a simple case, an FC may process only one ElementalSignal and that Signal may be supported by a single Strand that is by itself in a Cable. In this simple case there is clearly a 1:1 relationship between the FC and the Cable.

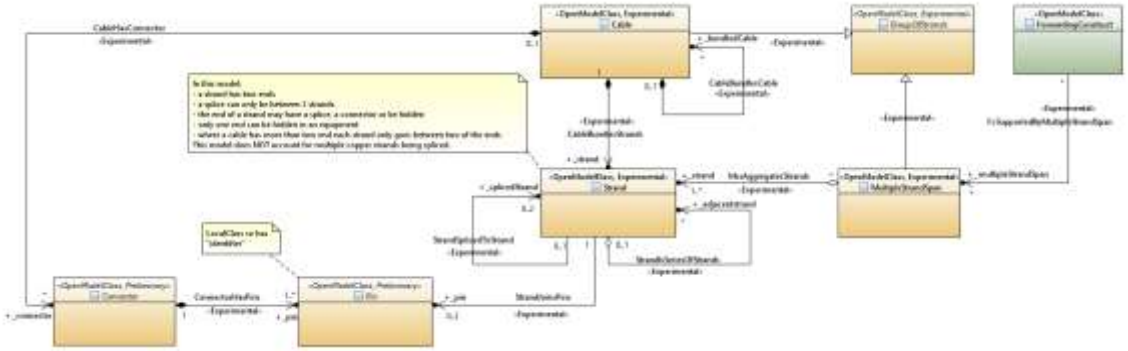
However, as noted earlier, this is a corner case. In a majority of cases, the FC carries more than one ElementalSignal and the relationship between the FC and the Strands of the Cable is not simple. Indeed, several signals may share, and be dependent on a single Strand or on several Strands. The groupings are such that if any Strands were to be removed from the group, then all the signals of the group would be lost, and such that the signals supported by GroupOfStrands are indivisible when they flow through the Strands.

To enable the necessary flexibility the Strands of the Cable are decoupled from the FC via the MultipleStrandSpan. The MultipleStrandSpan terminates on AccessPorts. The Strands in the group may be scattered across several Cables related to the way the Pins are scattered across several Connectors. However, it is likely that the Cable bundling of Cables may significantly reduce the spread of FCs across Cables compared to the spread of LTPs across Connectors (a Cable may have many Connectors and a Cable my bundle Cables). This means that the FCs carrying an indivisible signal will often travel along a single Cable through a single duct[3].

[2] This model does not cover full inside/outside plant modelling.
[3] The duct is not modelled as outside plant is beyond the scope of the model currently.

The GroupOfStrands is such that the FC depend upon "all" Strands in the group. An FC may itself be a grouping of FCs such that if any Strand is removed this would cause failure in signals such that the operation of the grouping FCs would be impacted (fail or degrade in some way).

The notes shown on the figure touch on some ongoing discussions.

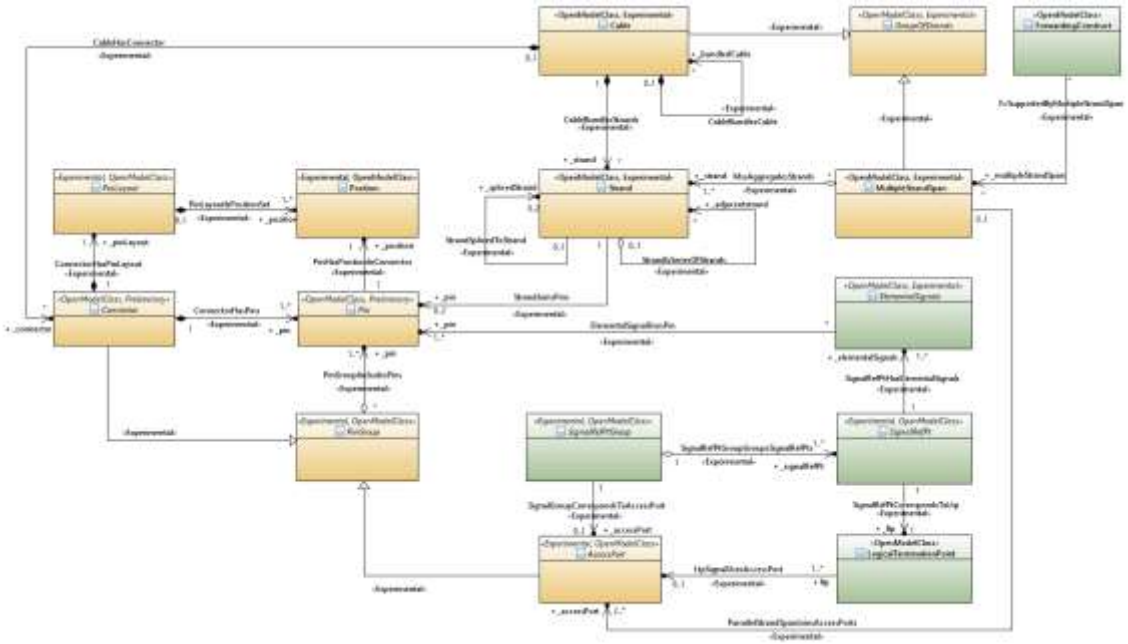


CoreModel diagram: Equipment-ConnectorPinCableStrandAndFc

Figure 3-6 Cable to FC

3.3.3 AccessPort and MultipleStrandSpan

The following figure shows both aspects of the model.



CoreModel diagram: Equipment-ConnectorPinCableStrandLtpAndFc

Figure 3-7 Connector/Cable to LTP/FC

3.3.4 Detailed description of the model fragment

The following sections list the classes introduced in this section and not covered in the previous sections of this document. See section 3.9 Physical Connector and conceptual Port on page 45 for some pictorial examples of the interaction of some of the entities discussed here.

3.3.4.1 AccessPort

Qualified Name: CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::AccessPort

A group of pins that together support a signal group where any one pin removed from the group will prevent all signals of the signal group from flowing successfully.
This class is abstract.

Inherits properties from:

- PinGroup

This class is Experimental.

3.3.4.2 ElementalSignals

Qualified Name:
CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::ElementalSignals

The elemental (sub-atomic) parts of an "indivisible" signal where processing in the LTP is required to extract the elemental signals.
This class is abstract.

This class is Experimental.

3.3.4.3 GroupOfPins

Qualified Name:
CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::GroupOfPins

A group of pins from one or more connectors relevant for some purpose.
This class is abstract.

This class is Obsolete.

3.3.4.4 GroupOfStrands

Qualified Name:
CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::GroupOfStrands

A group of strands from one or more cables relevant for some purpose
This class is abstract.

This class is Experimental.

3.3.4.5 MultipleStrandSpan

Qualified Name:
CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::MultipleStrandSpan

An adjacency between AccessPorts.
The adjacency is supported by a group of strands between pins of the AccessPorts.
This is a physical abstraction.

Inherits properties from:

- GroupOfStrands

This class is Experimental.

3.3.4.6 Pin

Qualified Name: CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::Pin

An individual physical connection point (male or female) that is not relevantly divisible.
May be capable of carrying electrical or optical signals.
A pin normally has only one wire/fiber strand attached.
It may have more than one wire/fiber attached but is such that the attachment forms a physical merge (all attached things receive exactly the same signal and any inputs to the pin are electrically/optically merged).
This class is abstract.

This class is Preliminary.

3.3.4.7 PinGroup

Qualified Name: CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::PinGroup

A group of pins relevant for some purpose.
This class is abstract.

This class is Experimental.

3.3.4.8 PinLayout

Qualified Name: CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::PinLayout

The structuring of pins in a connector.
This class is abstract.

This class is Experimental.

3.3.4.9 SignalRefPt

Qualified Name:
CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::SignalRefPt

A single coherent signal as processed by a single LTP.
This class is abstract.

This class is Experimental.

3.3.4.10 SignalRefPtGroup

Qualified Name:
CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::SignalRefPtGroup

A conceptual access for a group of signals (where that group of signals cannot be separated).
A physical indivisible group of signals.
This class is abstract.

This class is Experimental.

3.3.4.11 Strand

Qualified Name: CoreModel::CorePhysicalModel::ConnectorAndPin::ObjectClasses::Strand

A Strand represents a continuous long, thin piece of a medium such as glass fiber or copper wire.

In this model a Strand:
- a strand has two ends
- a splice can only be between 2 strands.
- the end of a strand may have a splice, a connector or be hidden
- only one end can be hidden in an equipment
- where a cable has more than two end each strand only goes between two of the ends
This model does NOT account for multiple copper strands being spliced.
This class is Experimental.

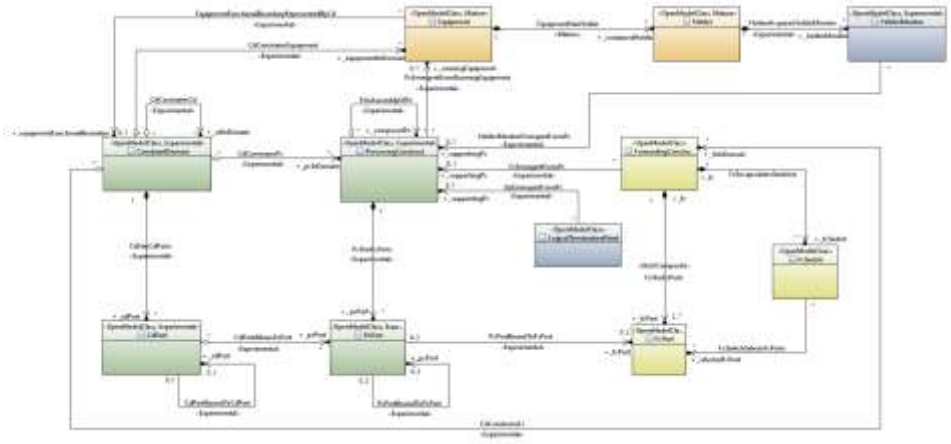
3.4 Equipment to Function Sketch

As ProcessingConstruct (see TR-512.11) was added in release 1.3 of the model and the Equipment to Function model was updated. Further analysis have improved some aspects of this model.

3.4.1 Equipment to Function using ProcessingConstruct and ConstraintDomain

In principle, functionality is emergent from equipment when it is powered. In the model, there are several representations of functionality (PC, LTP etc). Understanding the relationship between relevant network functions and the physical environment is relevant for many management-control activities including network planning, fault localization and engineering works impact analysis.

The following figure shows the use of PC to model the functionality emergent from an Equipment assembly.



CoreModel diagram: Equipment-ProcessingConstructAndResilience

Figure 3-8 Equipment to Function using ProcessingConstruct and showing Resilience

The axis of emergent functionality are the PC and CD. As shown in the figure above, a CD can be used to define the functional boundary of one[4] or more equipments (EquipmentFunctionalBoundaryRepresentedByCd). At the base of relevant functional emergence is a PC that relates to one single equipment (PcEmergentFromRunningEquipment). PCs may be assembled into systems of PCs interconnected through their PcPorts (PcPortBoundToPcPort) that can then be viewed as PCs (PcIsAssemblyOfPc). A PC can also be broken down into alternative subordinate parts (also via PcIsAssemblyOfPc).

Hence, the ProcessingConstruct is used to represent:

- The function blocks supported by an Equipment
- The decomposition of each function block into atomic parts
- The assembly of the atomic parts into more aggregate functions (across Equipment boundaries potentially) that can be seen as function blocks for further decomposition and aggregation as appropriate

At some point in this complex recursion a relevant network function is emergent (e.g. LtpIsEmergentFromPc[5]).

The "PcIsAssemblyOfPc" is essentially a compact form of the Component-System pattern (see TR-512.A.2).

3.4.2 "Equipment Protection"

A combination of Equipment can offer protected functions. This is not protection of the equipment as the individual Equipments are not in themselves physically resilient. It is also not protection of the entire capability of an equipment as some functions on an Equipment may be protected by functions on another equipment in one way, some in another way and some may not be protected.

The model enables the representation of the necessary degrees of protection. The essential protection is supported in terms of the PC where a PC instance may represent a function on one equipment, another PC instance may represent the same function on another equipment and a further PC instance represents the resilient function resulting from the appropriate combination of the two.

The model allows different levels of precision:

1. Basic representation via PcIsAssemblyOfPc such that a resilient PC is an assembly of non-resilient PCs
2. More explicit representation via PcPortBoundToPcPort where the functionality to be protected is exposed at a port of an instance of a PC on one equipment and at a port of an instance of the same type of PC on another equipment and these two ports are bound to ports of a PC that exposes a resilient form of the function function (i.e., the 0..2 cardinality of PcPortBoundToPcPort as shown in the previous figure above).
3. Detailed and generalized via PcPortBoundToFcPort where the same functions noted in (2) are not directly bound but are bound via an FC that allows exposure of the FcSwitch and Casc and other resilience properties in a standard way. This model can be used for complex protection schemes

[4] Clearly, the CD may have some other purpose and hence the association is multiplicity "*".
[5] This association was not navigable from the LTP in 1.3.1 but is now navigable.

The figure below shows two instances of function A on different equipments being combined to form a resilient function Ar.

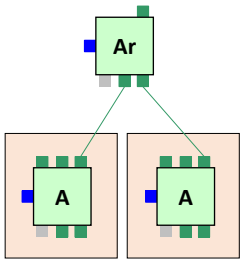


Figure 3-9 "Equipment Protection"

See also Figure 4-3 Simplified sketch of forms of resilience in an NE.

3.4.3 Classes associated with Equipment to function mapping with resilience

The following is a summary of the key classes and attributes related to Equipment to function mapping with resilience.

3.4.3.1 ProcessingConstruct (PC)

This section focuses on the aspects that support functional decomposition, functional aggregation/assembly and functional abstraction.

Qualified Name: CoreModel::ProcessingConstructModel::ObjectClasses::ProcessingConstruct

ProcessingConstruct (PC) can be used to represent both potential and enabled processing between two or more of its PcPorts.
A PcPort may be associated with another PcPort or with an LTPs at a particular specific layerProtocol.
Like the LTP, the PC supports any transport protocol including all circuit and packet forms.
The PC is used to effect processing of information extracted from the transport layer protocol signal.
A PC may be:
- Asymmetric such that it offers several functions and such that different functions are offered to different attached entities.
- Symmetric such that it offers (or is considered as offering) only one function and the same function is offered to any attached entity with no interactions between functions offered to each attached entity
An asymmetric PC offering a number of distinct functions will have PcPorts through which the distinct functions are accessed.
A symmetric PC offering only a single function need not have PcPorts, the function can be accessed directly from the PC.

Inherits properties from:

- GlobalClass

This class is Experimental.

Table 23: Attributes for ProcessingConstruct

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_pcPort	Experimental	An asymmetric PC instance is related to LTPs via PcPorts (essentially the ports of the PC). Symmetric PCs don't have PcPorts and are directly related to LTPs.
_composedPc	Experimental	The PC class supports a recursive aggregation relationship (PcIsAssemblyOfPc). This allows both: - abstraction where an assembly of PCs (forming a System) is viewed as an abstract PC - decomposition such that the internal construction of a PC can be exposed as multiple lower level PCs. Appropriate use of this association allows each of a collection of PCs to be decomposed into atomic parts (PCs) that can then be assembled into set of complex functions where each function in the set can be viewed as a PC. Note that the model actually represents an aggregation of lower level PCs into higher level PCs as viewpoints rather than partitions, and supports multiple views.

3.4.3.2 PcPort

This section focuses on assembly of components with no need for LTPs where the component association in the assembly is direct and indirect via a resilience mechanism.

Qualified Name: CoreModel::ProcessingConstructModel::ObjectClasses::PcPort

Represents the access to the functionality of a PC.

Inherits properties from:

- LocalClass

This class is Experimental.

Table 24: Attributes for PcPort

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_pcPort	Experimental	A PcPort can be directly bound to another PcPort (rather than via a LTP) to support a simplified application level view (rather than requiring the full transport level view).
_fcPort	Experimental	A PcPort can be directly bound to an FcPort (rather than via a LTP) to support a simplified application level view (rather than requiring the full transport level view). This is used to represent complex semantic associations between PCs where _pcPort direct association is not sufficient.

3.4.3.3 ForwardingConstruct (FC)

This section focuses on the protection aspects of the FC.

Qualified Name: CoreModel::CoreNetworkModel::ForwardingConstruct::ForwardingConstruct

The ForwardingConstruct (FC) represents enabled constrained potential for forwarding between two or more FcPorts at a particular specific layerProtocol .
The constraint is explained by the FcSpec. Even when an FC is in place enabling potential for flow, it is possible that no information is flowing as there is no flow matching the constraint, hence "potential".
Like the LTP, the FC supports any transport protocol including all analogue, circuit and packet forms.
The FC is used to effect forwarding of transport characteristic (layer protocol) information.
An FC can be in only one ForwardingDomain (FD).
The FC is a forwarding entity.
At a low level of the recursion, a FC represents a cross-connection within an NE. It may also represent a fragment of a cross-connection under certain circumstances.
The FC object can be used to represent many different structures including point-to-point (P2P), point-to-multipoint (P2MP), rooted-multipoint (RMP) and multipoint-to-multipoint (MP2MP) bridge and selector structures for linear, ring or mesh protection schemes.
When applied to media, the FC represents the ability for a flow/wave (potentially containing information), to be propagated between FcPorts.
The existence of a FC instance is independent of the presence (or absence) of a flow/wave (and any information encoded within it) where flow/wave covers the progressing of any analogue or digital (packet/frame etc.) structure.
A flow/wave cannot propagate in the absence of a FC instance.

Inherits properties from:

- ForwardingEntity
- GlobalClass

Table 25: Attributes for ForwardingConstruct

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_fcPort		The FcPorts define the boundary of the FC. The FC is accessed via the FcPorts. Flow within the FC is defined in terms of its FcPorts.
_fcSwitch		If an FC exposes protection (having two FcPorts that provide alternative identical inputs/outputs), the FC will have one or more associated FcSwitch objects. The arrangement of switches for a particular instance is described by a referenced FcSpec.

3.4.3.4 FcSwitch

Qualified Name: CoreModel::CoreNetworkModel::ForwardingConstruct::FcSwitch

The FcSwitch class models the switched forwarding of traffic (traffic flow) between FcPorts and is present where there is protection functionality in the FC.
If an FC exposes protection (having two or more FcPorts that provide alternative identical inputs/outputs), the FC will have one or more associated FcSwitch objects to represent the alternative flow choices visible at the edge of the FC.
The FC switch represents and defines a protection switch structure encapsulated in the FC and essentially "decorates" FCs that are involved in resilience schemes that use switching in a protection mechanism.
Essentially FcSwitch performs one of the functions of the Protection Group in a traditional model. It associates 2 or more FcPorts each playing the role of a Protection Unit.
One or more protection, i.e. standby/backup, FcPorts provide protection for one or more working (i.e. regular/main/preferred) FcPorts where either protection or working can feed one or more protected FcPort.
The switch may be used in revertive or non-revertive (symmetric) mode. When in revertive mode it may define a waitToRestore time.
It may be used in one of several modes including source switched, destination switched, source and destination switched etc. (covering cases such as 1+1 and 1:1).
It may be locked out (prevented from switching), force switched or manual switched.
It will indicate switch state and change of state.
The switch can be switched away from all sources such that it becomes open and hence two coordinated switches can both feed the same LTP so long as at least one of the two is switched away from all sources (is "open").
The ability for a Switch to be "high impedance" allows bidirectional ForwardingConstructs to be overlaid on the same bidirectional LTP where the appropriate control is enabled to prevent signal conflict.
This ability allows multiple alternate routes to be present that otherwise would be in conflict.

Inherits properties from:

- LocalClass

Table 26: Attributes for FcSwitch

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_selectedFcPort		Indicates which points are selected by the switch. Depending on the switch spec (via FcSpec) - more than one FcPort can be selected at any one time (e.g. egress switch, ingress packet switch) - zero FcPorts can be selected. For an ingress switch this indicates that the switch common (egress) is "high impedance" .

3.4.3.5 Equipment

This section focuses on physical support for functionality.

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentPatternStructure::ObjectClasses::Equipment

Represents any relevant physical thing.
Can be either field replaceable or not field replaceable.
Note: The model is currently constrained to inside plant.

Inherits properties from:

- GlobalClass

This class is Mature.

Table 27: Attributes for Equipment

Attribute Name	Lifecycle Stereotype (empty = Mature)	Description
_equipmentFunctionalBoundary		See referenced class

3.4.4 V1.4 refinements of V1.3 model

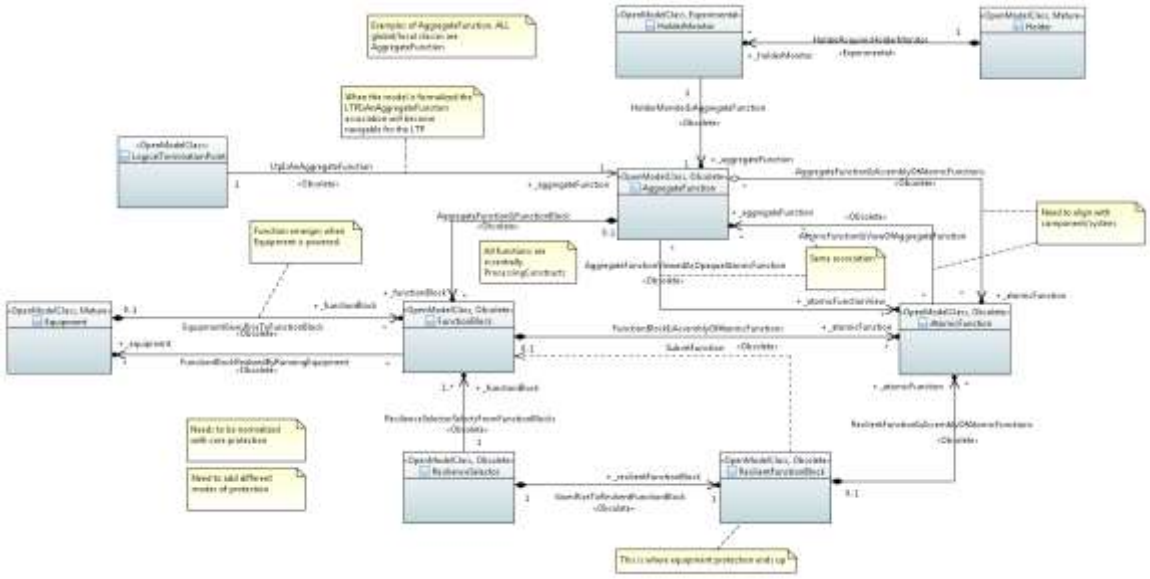
In release 1.3 there were two direct associations between PC and Equipment. These associations have been normalized with those from the Software model (see TR-512.12)

These have been replaced by a, more versatile but less apparent, use of ConstraintDomain.

3.4.5 Obsolete model from V1.2 use to illustrate the V1.3 model

The figure below is a sketch model relating Equipment to Function from V1.2. Whilst the classes are obsolete they still provide a helpful illustration of the model. As the comment in the figure says, "All functions are essentially ProcessingConstruct" (i.e. any class with the word "function" in its name is replaces by PC in the V1.3 model described above). The Equipment functionality support is first exposed as coarse FunctionBlocks (e.g. arithmetic process, traffic process etc). The function block may be made resilient via a complex protection switch which can select the functionality from one or more instances[6]. The FunctionBlock is then decomposed into AtomicFunctions which can then be assembled to form AggregateFunctions (e.g. the LTP). These two steps allow versatile mapping from the hardware-oriented function blocks to the conceptual functions such as LTP and LayerProtocol where the conceptual function may be "smeared" across several FunctionBlocks. The model also allows for recursive decomposition and assembly to any depth to allow for cases where intermediate representations are necessary to describe the functional emergence. For some illustrative figures of this see section 4.2 Physical to functional model on page 48.

[6] The protection model has only had very limited development so far and the model is clumsy in this area.



CoreModel diagram: Equipment-ObsoleteEquipmentToFunction

Figure 3-10 Equipment to Function

3.4.5.1 AggregateFunction

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentToFunction::ObjectClasses::AggregateFunction

Represents some assembly of atomic functions that can be considered as useful from some perspective. Can be viewed as one or more functional blocks (essential leading to a recursive cycle of Block --> Atomic --> Aggregate --> Block).
Each of the functional entities in the model is a view of a single AggregateFunction.
This class is Obsolete.

3.4.5.2 AtomicFunction

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentToFunction::ObjectClasses::AtomicFunction

Represents the micro-function that is the largest function of the functional block that will not need to be subdivided when forming the relevant abstract views (i.e., it can just be assembled).
This class is Obsolete.

3.4.5.3 FunctionBlock

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentToFunction::ObjectClasses::FunctionBlock

Represents the chunks of base functionality provided by the equipment.
The chunks of base functionality are likely to relate to the hardware layout and be quite distinct from the functions of the familiar abstract representation.
The functions are necessarily abstract and, to a degree, virtualized.
This class is Obsolete.

3.4.5.4 ResilienceSelector

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentToFunction::ObjectClasses::ResilienceSelector

Represents the ability to select capability from two or more identical FunctionalBlocks so as to give rise to an equivalent emergent resilient function.
This class is Obsolete.

3.4.5.5 ResilientFunctionBlock

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentToFunction::ObjectClasses::ResilientFunctionBlock

Represents the functions emergent from a function protection process.
The emergent functions are necessarily significantly virtualized.
This class is Obsolete.

3.5 FRU and non-FRU

Note that the approach in this subsection to representation of the constraints using inheritance with stereotype is highly experimental. It is likely that this approach will change as the model progresses.

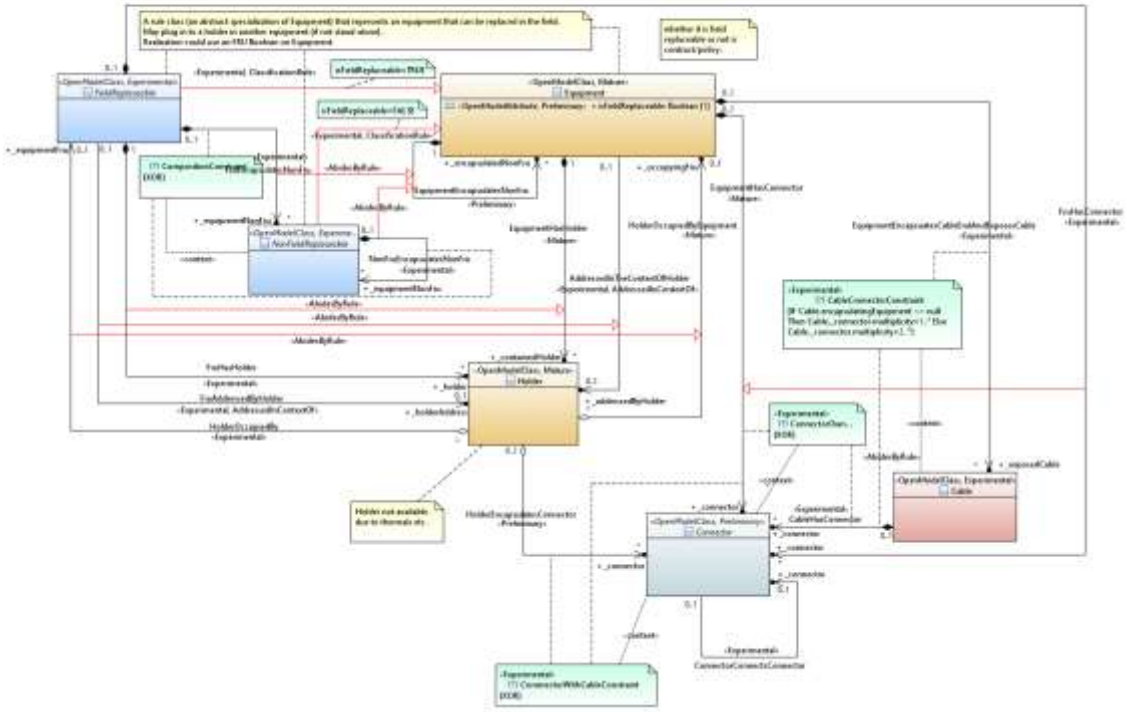
Considering the distinction between FRU and nonFRU there were two choices, model the FRU explicitly (and then expect instantiation of instances of distinct FRUs and non-FRUs classes) or model the FRU and non-FRU as Equipment with solely an attribute to indicate the case. The attribute approach was preferred but that lost rule detail present in the distinct class case.

So as not to lose the constraints, the FRU/non-FRU class distinctions were kept, those classes were made abstract and then applied as shown to the attribute based model developing an experimental technique

This technique is to use generalization modulated with stereotypes to represent the narrowing of a class to cover a defined case. The narrowed class does not gain attributes, the general class is fully populated whereas in the narrowed class attributes take specific fixed values where the specializations are all abstract and the generalization is concrete. The aim is to develop machine interpretable rule systems that allow the behavior of an instance of a generalized class to be constrained based upon the case.

The experimental stereotype « ClassificationRule » carries the properties that define the case, the stereotype « AbidesByRule » identifies the generalized association that constrained by the specialized association.

This model is experimental and requires significant further development. It is likely that an alternative form will eventually be used.



CoreModel diagram: Equipment-EquipmentToHolderRules

Figure 3-11 FRU and Non-FRU rules

3.5.1 FieldReplaceable

Qualified Name:
CoreModel::CorePhysicalModel::RuleModels::FruNonFruRules::ObjectClasses::FieldReplaceable

A rule class (an abstract specialization of Equipment) that represents an equipment that can be replaced in the field.
May plug in to a holder in another equipment (if not stand-alone).
Realization could use an FRU Boolean on Equipment.
This class is abstract.

Inherits properties from:

- Equipment

This class is Experimental.

3.5.2 NonFieldReplaceable

Qualified Name:
CoreModel::CorePhysicalModel::RuleModels::FruNonFruRules::ObjectClasses::NonFieldReplaceable

A rule class (an abstract specialization of Equipment) that represents an equipment that cannot be replaced in the field.
Is simply a subordinate part of an FRU (or another NFRU – where there must be an FRU at the top of the hierarchy).
Does not have any exposed holders (any associated holders are assumed to belong to the containing FRU).
Does not have any connectors (any associated connectors are assumed to belong to the containing FRU).

This class is abstract.

Inherits properties from:

- Equipment

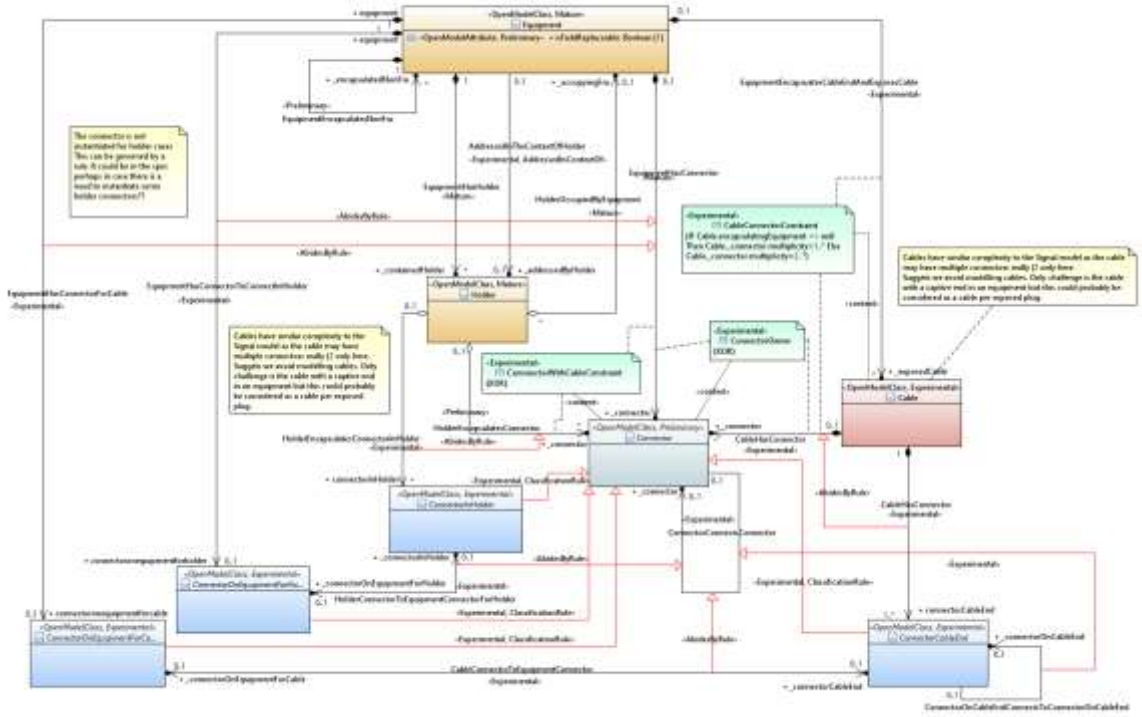
This class is Experimental.

3.6 Connector Rules

Note that the approach in this subsection to representation of the constraints using inheritance with stereotype is highly experimental. It is likely that this approach will change as the model progresses.

Similar to the previous section, the figure below shows an experimental method for representation of restrictions in the model. The figure below shows a representation of the rules for the exposure of Connectors on an Equipment accounting for both the FRU/Non-FRU differences and also for the differences between Connectors related to a Holder versus visible Connectors available to connect cables to.

Essentially the figure shows that Connectors related to Holders have different lifecycles and visibilities than Connectors related to cables.



CoreModel diagram: Equipment-ConnectorRules

Figure 3-12 Connector rules

3.6.1 ConnectorCableEnd

Qualified Name:
CoreModel::CorePhysicalModel::RuleModels::ConnectorRules::ObjectClasses::ConnectorCableEnd

A rule class (an abstract specialization of Connector) that represents a connector on the end of a cable.
This class is abstract.

Inherits properties from:

- Connector

This class is Experimental.

3.6.2 ConnectorInHolder

Qualified Name:
CoreModel::CorePhysicalModel::RuleModels::ConnectorRules::ObjectClasses::ConnectorInHolder

A rule class (an abstract specialization of Connector) that represents a connector that are only accessible to an equipment inserted in the holder.
This class is abstract.

Inherits properties from:

- Connector

This class is Experimental.

3.6.3 ConnectorOnEquipmentForCable

Qualified Name:
CoreModel::CorePhysicalModel::RuleModels::ConnectorRules::ObjectClasses::ConnectorOnEquipmentForCable

A rule class (an abstract specialization of Connector) that represents a connector exposed on an equipment such that a cable may be plugged in.
This class is abstract.

Inherits properties from:

- Connector

This class is Experimental.

3.6.4 ConnectorOnEquipmentForHolder

Qualified Name:
CoreModel::CorePhysicalModel::RuleModels::ConnectorRules::ObjectClasses::ConnectorOnEquipmentForHolder

A rule class (an abstract specialization of Connector) that represents a connector on an equipment that is intended to mate with a connector in a holder.
This class is abstract.

Inherits properties from:

- Connector

This class is Experimental.

3.7 Expected and Actual

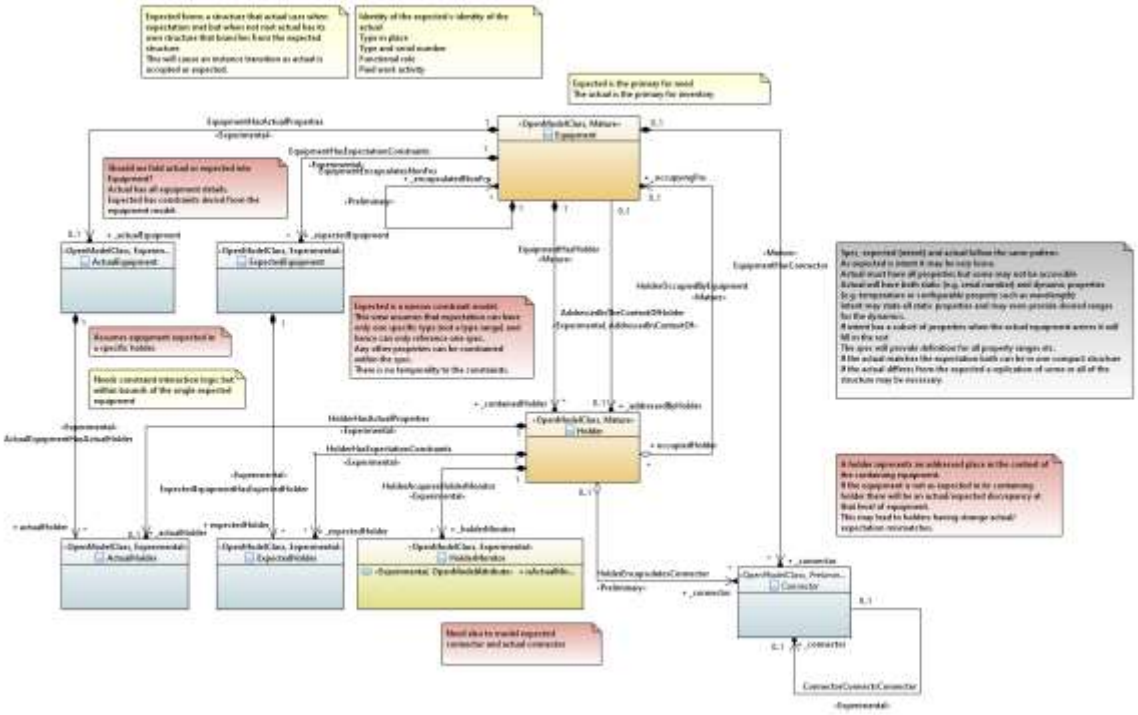
This model fragment explores a representation of expected and actual where the Equipment pattern is augmented with composed parts representing the expected and actual settings.

The ExpectedEquipment is likely to be in terms of constraints, with many "don't care" values (such as serial number) but is assumed to have a single precise definition of type and of position with respect to the Equipment/Holder it is to be contained in[7].

[7] In a client-provider context Expectation is the client view of an agreement that the provider intends to satisfy. As it is not possible to remotely install a physical thing there is a decoupling between the request and the actual realization. This is very similar to the decoupling at a contractual interface where there is an agreement to provide something and hence an intention by the provider and an expectation from the client. The agreement is satisfied once the provider provides what was agreed. In the general case any properties could be constraints. In the case of the Equipment the agreement is more precise providing no choice of positioning. It is possible under some circumstances that position flexibility may also be allowed. This aspect is for further study.

The assumption is that when there is a mismatch, the expected Equipment will have a set of expected Holders and the actual Equipment a potentially overlapping set of actual Holders. The hierarchy is driven by Holder position.

Where the Equipment is stand-alone (not in a Holder) it is assumed that geographical location or other statement of place will enable detection of expectation/actual mismatch. Hence stand-alone Equipment can also have an expected and actual value. Also see section 4.3 Actual v expected on page 50 for a pictorial example of work in progress.



CoreModel diagram: Equipment-ExpectedAndActual

Figure 3-13 Expected and actual

3.7.1 ActualEquipment

Qualified Name:
CoreModel::CorePhysicalModel::ExpectedAndActual::ObjectClasses::ActualEquipment

The equipment that is actually present in the physical network.
It will expose all dynamic properties and some critical static properties.

Inherits properties from:

- EquipmentDetail

This class is Experimental.

3.7.2 ActualHolder

Qualified Name:
CoreModel::CorePhysicalModel::ExpectedAndActual::ObjectClasses::ActualHolder

A holder in the ActualEquipment.

Inherits properties from:
- HolderDetails

This class is Experimental.

3.7.3 ExpectedEquipment

Qualified Name:
CoreModel::CorePhysicalModel::ExpectedAndActual::ObjectClasses::ExpectedEquipment

A definition of the restrictions on the equipment that is expected to be present in the physical network at a particular "place".
The expected equipment will state the type and may constrain any other invariant properties. It may also provide desired ranges for dynamic properties.

Inherits properties from:
- EquipmentDetail

This class is Experimental.

3.7.4 ExpectedHolder

Qualified Name:
CoreModel::CorePhysicalModel::ExpectedAndActual::ObjectClasses::ExpectedHolder

A definition of a holder expected in the ActualEquipment (i.e. an ActualHolder) as part of the constraints provided by the ExpectedEquipment.

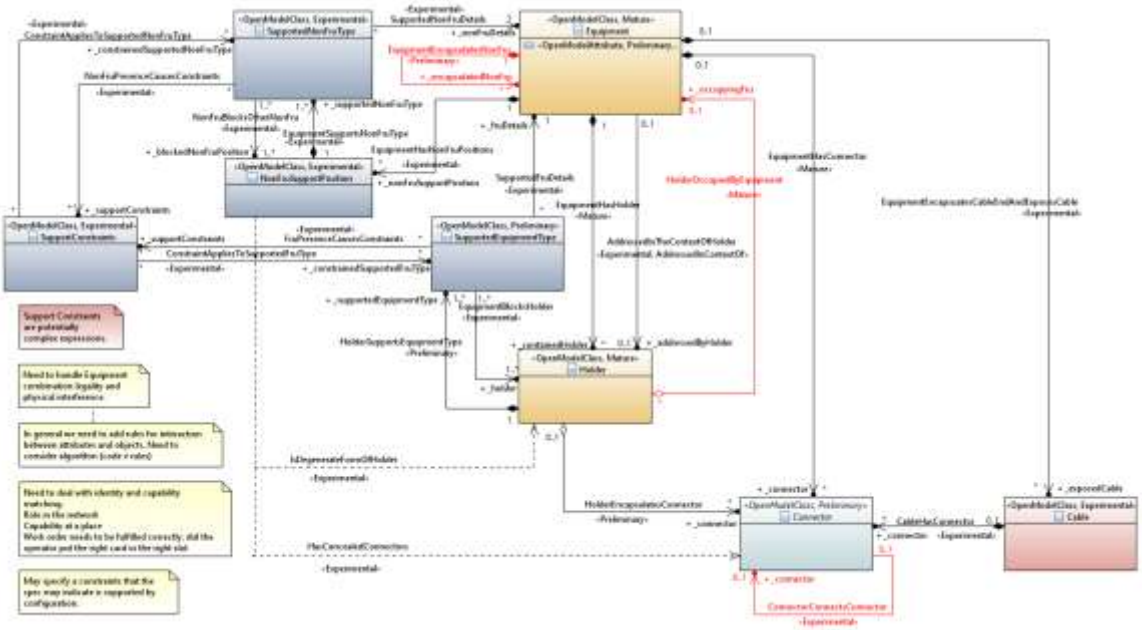
Inherits properties from:
- HolderDetails

This class is Experimental.

3.8 Specification

The figure below provides a fragment of a candidate specification model for Equipment, focusing on Holder compatibility and supported non-FRUs.

Note that the EquipmentEncapsulatesNonFru association and the HolderOccupiedByEquipment association (both shown in red) are essentially governed by the rules stated in the supportedNonFruType class and SupportedEquipmentType class (and their associated classes) respectively.



CoreModel diagram: Equipment-ConstraintsOnEquipmentPattern

Figure 3-14 Specification

3.8.1 NonFruSupportPosition

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentSpecification::ObjectClasses::NonFruSupportPosition

Equivalent to the holder for the FRU, represents in the specification a place where one or more types of non-FRU could be present.
Unlike the FRU in a Holder, the non-FRU present is fixed in place whilst the equipment is in the field (as it is not Field-replaceable).
This class is Experimental.

3.8.2 SupportConstraints

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentSpecification::ObjectClasses::SupportConstraints

Rules related to how both non-FRU and FRU presence restricts the potential for additional equipments to be installed.
An FRU type installed in one holder may limit the FRU types that can be installed in another holder etc.
This class is Experimental.

3.8.3 SupportedEquipmentType

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentSpecification::ObjectClasses::SupportedEquipmentType

The FRU equipment types supported by the holder.
This class is Preliminary.

3.8.4 SupportedNonFruType

Qualified Name:
CoreModel::CorePhysicalModel::EquipmentSpecification::ObjectClasses::SupportedNonFruTyp
e

The non-FRU equipment types supported by the non-FRU support position.
This class is Experimental.

3.9 Physical Connector and conceptual Port

The Connector and Port are modelled as distinct abstract entities. It is likely that in both cases only the "name" is required in an instance realization. Both the Port and Connector represent fixed rules of grouping and these rules could be completely contained in the spec. The Connector also has physical properties but again these are fixed and could be contained in the spec.

An LTP instance provides an AccessPort reference. The AccessPort represents a PinGroup that relates to the Connectors via Pins as described in the appropriate spec. The Connector is part of an Equipment. An Equipment instance references a spec that identifies the Connectors.

There are many potential arrangements of association of LTP to AccessPort, Pin and Connector. The following sequence of figures provides a view of some of the variety.

As noted previously the AccessPort is a grouping of pins related to some coherent traffic flow. As can be seen from the sequence of figures there is no fixed relationship from AccessPort to Connector or Pin.

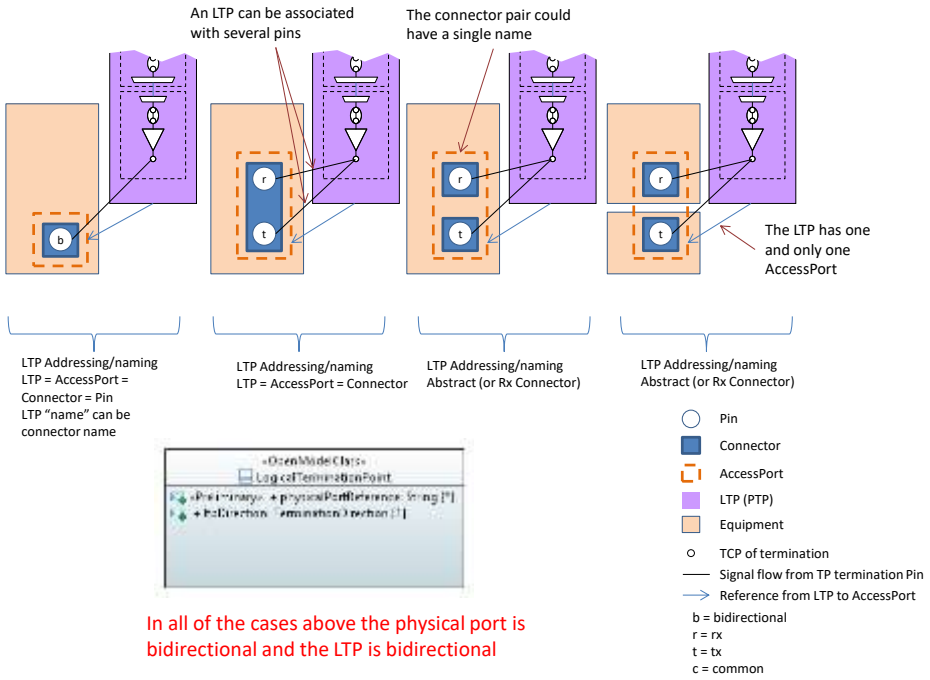


Figure 3-15 Basic cases of Access Port Reference

The figure above shows some simple LTP – AccessPort – Pin – Connector cases. The diagram case on the left could be where a single fiber is being used to convey a bidirectional signal (a coupler/splitter is within the Equipment) and hence only a single pin is required on a single Connector. In this case the Connector, Pin, AccessPort and LTP all have multiplicity [1]. The diagram case on the right could be again an optical case (with one pin per Connector) where the LTP is being considered as bidirectional but there are separate dedicated Equipments for each direction of traffic.

The pin of the Connector is essentially omni-directional (i.e. the media has no directional limitation and the material would allow the photons/electrons etc to propagate in any direction – see documentation on media in TR-512.2 and TR-512.A.4). The construction of the Connector (and the associated Strand of material) does in some cases provide such a narrow channel that the propagation is limited to essentially two directions but even in these cases the material is essentially omni-directional. The directionality designation associated with a Connector is "inherited" from the directionality of the termination function attached. In the figure above/below the rx and tx designations result from the attached terminations.

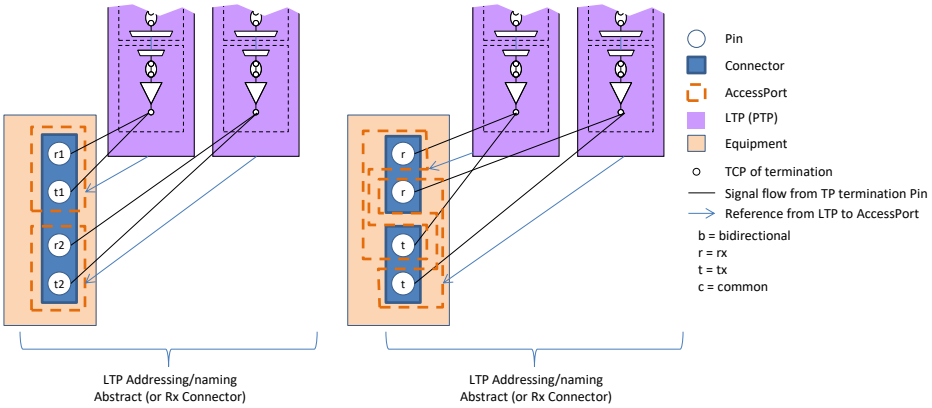


Figure 3-16 More Complex cases of intertwined Connectors

The diagram on the left in the figure above could represent a case where there is ribbon cable with multiple (in this case four) fibers terminated with one Connector and where each fiber is being used for only one direction of signal (but the fiber is inherently omni-directional as noted above). The two LTPs shown are both bidirectional and hence use two Pins each. As the signal is bidirectional in nature the AccessPort is also bidirectional

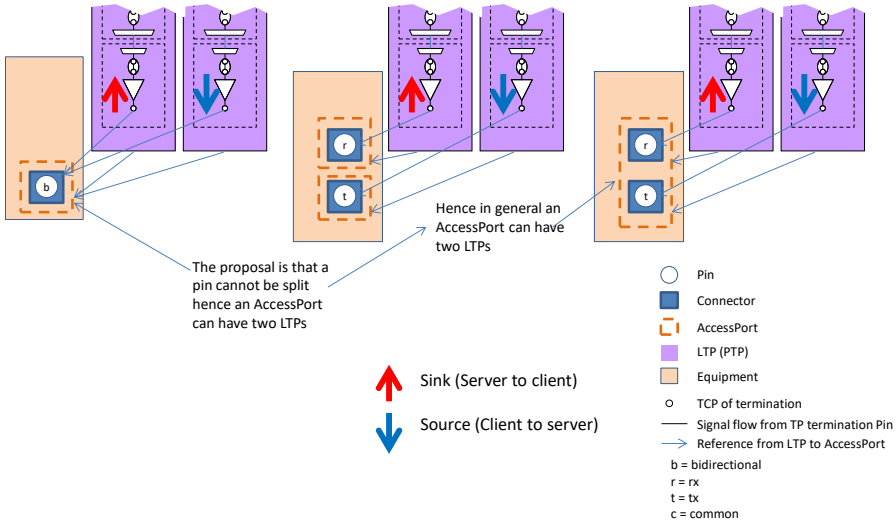


Figure 3-17 Unidirectional Cases

Other cases to consider:

- Multiple LTPs per AccessPort (as shown in the left and right cases in the diagram above)
- Multiple AccessPorts per pin

4 Work in progress (see also TR-512.FE)

4.1 Addressing

Traditionally ports have been identified using addressing schemes based on physical positioning, however there are challenges related to the complexity of potential spread across Equipments and ports as discussed in the previous section. The figure below discusses some cases. This work needs to be taken further.

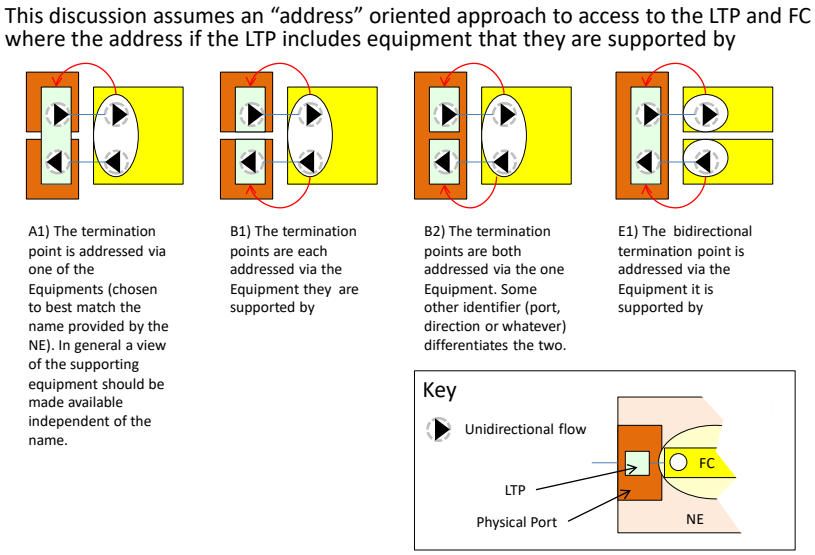
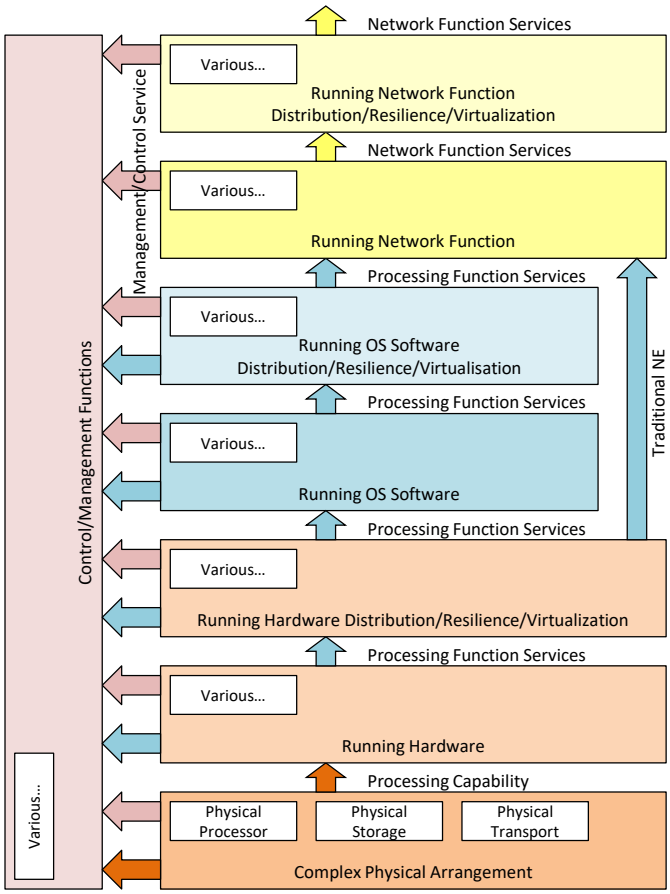


Figure 4-1 Connector/Port based addressing of LTPs

4.2 Physical to functional model

The figures in this section highlight aspects of the physical to functional relationship. The physical to functional model has been enhanced in V1.3 as a result of the addition of the Processing Construct (see TR-512.11). As a result, the cases set out below can now be supported with the model. The specific examples have not yet been included in the detailed documentation. As a consequence this section has not yet been removed.

The figure below shows a simplified hierarchy of functions emerging from running hardware/equipment where the emergent behavior considered as "function services". The figure is a rough sketch.



- Each block (Function, Software, Hardware, Physical and Control/Management) may be a single component or may be many components interacting to form one or more systems
- Each component may expose detail as if many components where the exposed components are interacting to form one or more systems
- Each set of services may be subdivided and may be complex reflecting the exposed components and their encapsulated functions
- Running OS Software is generalized to any OS including the negligible OS of a FPGA
- Distribution and resilience may not be present or may be present for only some of the services (management/control may also have distribution/resilience (not shown)
- A network function may be built directly on the Processing Function Services of the Running Hardware (i.e. is essentially a hardware realization) or some combination of running hardware and OS software
- Processing capability is a generalized term that could also apply to a parallel compute engine and a scattered set of dedicated engines such as any assembly of logic gates and buffers
- Each block is controlled/managed by a function that itself is built on Processing Capability and Processing Function Services
- The following figures use this colour scheme and often omit Management/Control

Figure 4-2 Simplified sketch of physical to functional

The figure below shows, very roughly, the emergence of various aggregate functions from functional blocks on two FRUs. In the figure:

- The FRUs are at the bottom (in brown)
- The inherent capabilities of the FRUs are shown as dotted boxes in the FRUs
- The emergent functional blocks are shown in green as are the aggregate functions
- A somewhat abbreviated progression from functional block to aggregate function via atomic function (not shown) to LTP (in blue at the top of the page) is shown
- The yellow bars represent different forms of protection essentially as FCs
- The functions in the yellow dotted shape are essentially virtual functions as the position of realization of the function is not fixed (and need not be known)

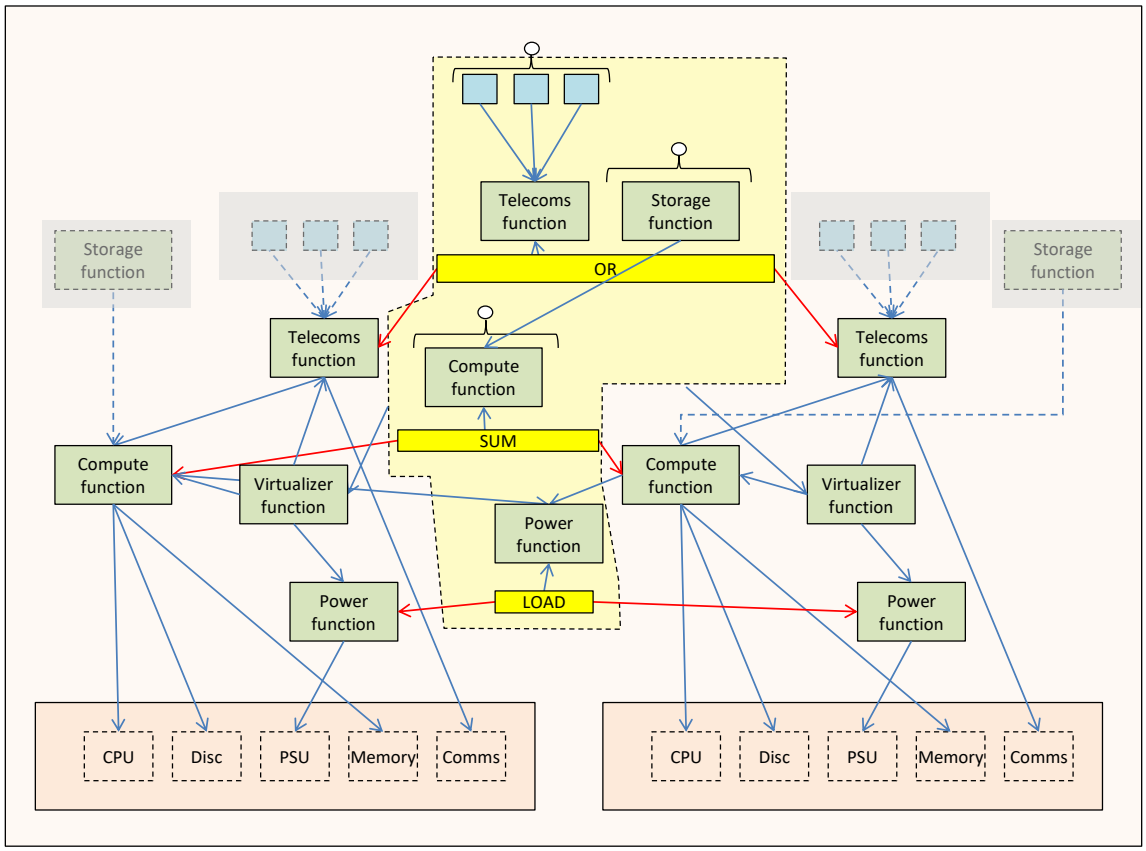


Figure 4-3 Simplified sketch of forms of resilience in an NE

4.3 Actual v expected

The figure below shows various cases of actual-expectation mismatch.

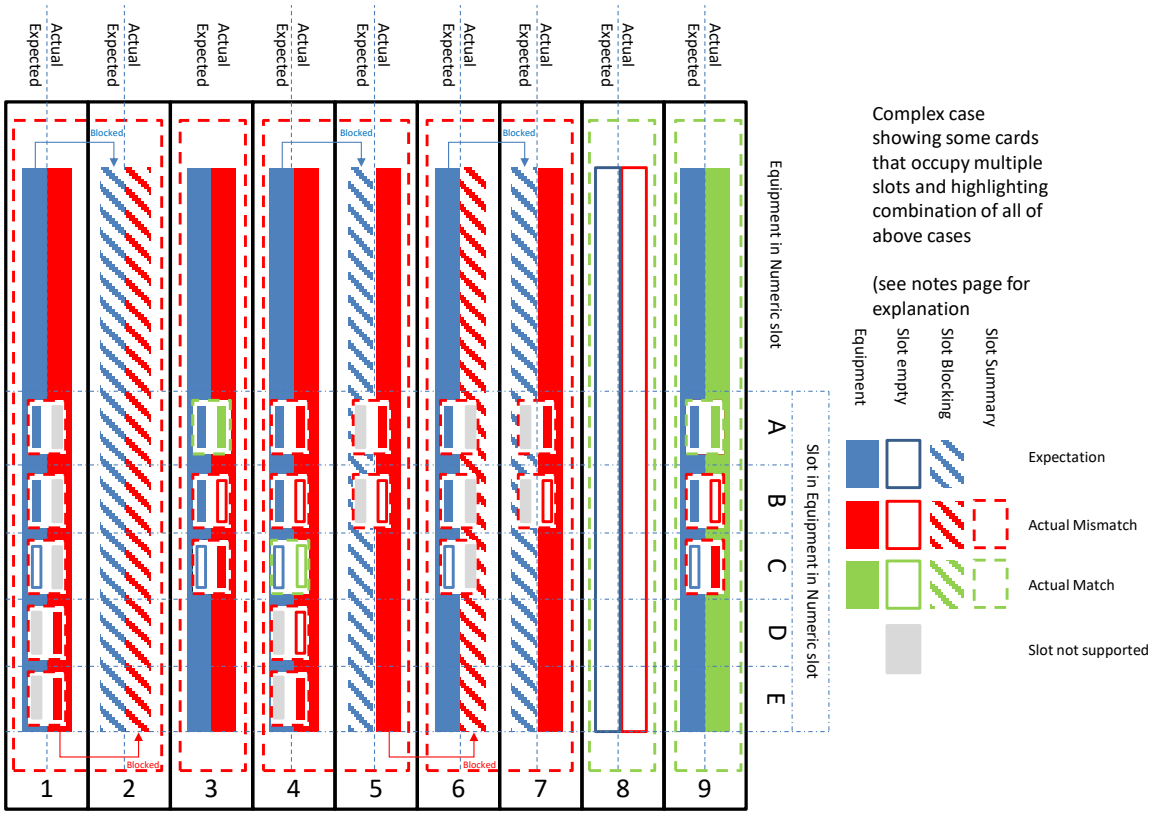


Figure 4-4 Expectation v actual showing mismatch and blocking

This figure above shows a representation of a SUBRACK (Equipment) with 9 SLOTS (Holders). Each Holder has a dotted line down the middle where the symbols in the slot to the left of the dotted line represent equipping expectation and to the right represent actual equipping. The large colored blocks represent the CIRCUIT_PACKs in the SLOTs (SLOT 8 is empty). The lower small colored rectangles represent the SLOTs in the CIRCUIT_PACKs (A – E) and the SMALL_FORMFACTOR_PLUGGABLEs (Equipment) in these SLOTS.

Considering the slots in turn

- Slot 1
  - Expectation: Double slot card with subslots (Holders) A-C (where A and B are expected to be equipped
  - Actual: Double slot card of different type with subslots D & E where both are actually equipped
  - Model: One Equipment object in slot 1 with 5 subslots A-E with 4 containing Equipment (two expected only and two actual only)
- Slot 2
  - Expectation: Blocked by expectation in slot 1
  - Actual: Blocked by actual in slot 1
  - Model: No Equipment
- Slot 3

    - Expectation: Single slot card with subslots A-C where A and B are expected to be equipped
    - Actual: Single slot card with subslots A-C where A is equipped matching expectation, B is not equipped (not matching expectation) and C is unexpectedly equipped
    - Model: One Equipment object in slot 3 with 3 subslots with each containing an Equipment
- Slot 4
    - Expectation: Double slot card with subslots A-C where A and B are expected to be equipped
    - Actual: Single slot card of different type with subslots A-E where A is equipped with the wrong card type, B is not equipped C is not equipped as expected etc
    - Model: One Equipment object in slot with 5 subslots with 3 containing Equipments
- Slot 5
    - Expectation: Blocked by expectation in slot 4
    - Actual: Double slot card with subslots A & B where A is actually equipped
    - Model: One Equipment object in slot with 2 subslots with 1 containing an Equipment
- etc
- Slot 9 has a CIRCUIT_PACK that matches the expectation (hence green actual column). It also has:
    - An SFP in slot A that matches expectation
    - Unexpectedly no SFP in slot B hence a mismatch
    - Unexpectedly an SFP in slot C hence the mismatch

Further explanation to be added in a later release.

The figure requires further development.

End of document