Core Information Model (CoreModel)

TR-512.A.15
Appendix – Controller Lifecycle and Security

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

List of Figures

Document History

Version	**Date**	**Description of Change**
1.6	January 2024	Initial Version

1 Introduction

This document is an appendix of the 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.

1.6 Appendix Overview

This document is part of the Appendix to TR-512. An overview of the Appendix is provided in TR-512.A.1.

2 Introduction to this Appendix document

This document considers the lifecycle of a system of Controllers. A Controller is an assembly of ControlConstructs etc. The model of control is discussed in TR-512.8.

3 Overview and Context

3.1 Business context

The discussion in this document focusses on future business opportunities to offer control services where those services enable a client to control, in a secure way, "slices" of network capability as if those slices were actual network.

The client perception of the control of the slice is that it seems no different to control of an owned network. The control pattern can be recursively applied and can be used to blend control of an owned network with that of slices, offered by another provider, in a seamless fashion.

It is recognized that some services of this form already exist, but it seems that there is no defined pattern for the development of these services through their lifecycle. This document explores control patterns for the development of these services.

The control patterns laid out are demonstrated as applied to network connectivity capability, but it appears that these patterns could apply to control of slices of any functional capability to any level of sophistication[1].

It is assumed in this document that there will be commercial justification to offer such services (in terms of revenue opportunity etc.). This document works through the technical aspects of this form of business interaction.

3.2 Management Control Architecture

It is assumed that the control architecture supports unified control of network structures, i.e., is not partitioned into vendor domains or by layer. This unified control offers a complete view of network capabilities.

Clearly, there is partitioning of control based upon commercial considerations, corporate regionalization, regulator constraints etc. The patterns developed in this document could apply at any of these boundaries. The document tends to consider the commercial boundary as this is probably the most onerous from a security perspective.

It is recognized that the control solution could be deployed on a mix of bare metal, private cloud and public cloud and some of the discussion in this document considers allocation of cloud resources. Not all possible arrangements are covered but none are considered excluded.

This document describes appropriate control structures and policy enforcement points to deal with the commercial boundary accounting for the interrelationship via the public internet and

---

[1] This could include control of compute/storage system, control of power systems, control of medical capabilities etc.

hence for threat actors within that environment. The document does not dig deeply into any details of security techniques etc. but does assume that state-of-the-art techniques will be used.

This document highlights appropriate control structures to enable the offering of control of slices to a client and to handle that control throughout its lifecycle.

3.3 Controller internal architecture

To provide the services considered, a particular arrangement of internal control structures is required. This document works through the lifecycle of emergence of these structures and explains their interaction.

In section 4 the initial formation of the control structure of a new business is considered and developed briefly to set a scene. The remainder of the document works through an overview of the necessary control elements and develops a picture through various stylized examples.

This document builds especially on models and discussions set out in TR-512.8 as well as aspects of other documents in the TR-512 set.

4 Formation of a Controller

The formation of a controller is considered from the beginning of the formation of a business. This is a general discussion that applies to any services. This section should be read with the focus of the control services in mind. Clearly, most businesses are at some intermediate stages of evolution. There are many challenges related to migration from a brown-field starting point that are not addressed by this discussion.

The initial controller is assumed to be one that deals with the growth of the business in a speculative environment, where the specific business purpose is forming through market exploration and resulting innovation. At this stage of growth, the control considerations relate to business strategy and development, where control actions are taken on more conceptual entities in the speculative business plan.

The early stages are skimmed over here to avoid philosophical debates. It is assumed that at some stage in the evolution, there is an understanding of at least some potential service offerings, and that at that stage these offerings have been designed to the point where they are considered for trial in some way and where offering them to prospective clients is considered as potentially valuable so that there is a desire to start negotiation for those services.

4.1 Forming the Service Definition and the Catalogue

The services will have been developed to some sufficient degree to be recognized as having relevant revenue potential, and appropriate rough designs of the service structure patterns, and alternative realization patterns will be available. Prior to offering in the market place the service definitions has to be such that what needs to be exposed to a potential client through negotiation is clearly separate from what needs to be available to the components of the controller such that it can determine the specific realization detail.

It is assumed here that the service offer will be of network capacity (the slice of the network) and that that network capacity will be defined in terms of a structure of ForwardingDomains interconnected by ForwardingConstructs of the appropriate form.

Clearly, not all possible services will be supported by the underlying network, hence the service offer will also include a set of forwarding service types that are allowed to be created in the slice. Even where a forwarding service type is supported, it may not be initially offered as there may be an opportunity for further revenue to be achieved by unlocking that capability via license added to the basic network slice etc. Unlocking additional potential is not considered in detail in this document.

In the ONF Core model, the forwarding service is defined by one or more specific forwarding construct(s) with associated constraints and as a consequence the forwarding service spec is an FcSpec. The network capability offer by the ForwardingDomain structure needs to be expressed in terms of FdSpecs[2].

The relationship between the external facing service definition[3] and the internal service definition[4] is essentially that between the component views in the component-system pattern (see

---

[2] For this to be fully available for use, the FcSpec and FdSpec will require further development.

[3] The TM Forum customer facing service.

TR-512.A.2). The service offered can be considered as a component, where the exposed capability of that component can be described in terms of a system of components and the realization of that component can be described in terms of a system of components[5].

The external facing definition provides a description of the effect of the service (component) offered. The internal facing definition provides a description of the realization.

The realization of a service will be such that there is a recursion of component-system decompositions.

The control service definitions, the key focus of this document, will be in terms of control specs[6] which will include ControlConstruct patterns and ControlTask patterns. The patterns will be essentially arrangements of Occurrences[7] of ControlConstructs, ControlTasks etc.

It is assumed that the operator will expose the structure of such services via some commercial interface and that that exposure will be considered as a catalogue.

The representation in the catalogue will necessarily include the pattern detail but will also need some abstract statements of capability and, especially, value that can be advertised so as to attract the potential clients in the first place.

From the perspective of realizing the service, there will be appropriate linkage from the internal description and the external description via ViewMappingFunctions (as discussed in this document and in TR-512.8).

4.2 Specific service assumptions

As noted above, the key consideration in this document is the deployment of control services. These services need to be defined in terms of control functionality and security from an external perspective and in terms of detailed control functionality, deployments and policy enforcement internally.

4.3 Progressing through the formation

The following sections work through the formation of controller structures and identify operational roles related to those formations. Various example service structures and corresponding roles are identified.

It is important to recognize that the roles identified may be human or machines roles. For example, in some early realizations of a solution an admin user role may be a person, whilst through some evolution, that admin user role may have been automated, potentially through application of AI.

---

[4] The TM Forum resource facing service.

[5] The exposed system description and the realization system description are clearly related but are not the same.

[6] The control spec has not been developed in detail but will follow the pattern of the other specs set out in TR-512.7.

[7] Occurrences, discussed in TR-512.7, are semi-specialized forms that appear in other patters and specifications.

5 Building a Controller

This section provides a brief overview of the steps to create an SDN controller including platform creation, the assignment of resources and the creation and configuration of the server contexts. It is assumed that the compute resources and the data communications network (DCN) are in place before a controller is created or modified. Three different administrator roles with different scope perform these steps. Note that one person could perform more than one administrator role, or an administrator role may be implemented by the application of a predefined policy, or an administrator role could be performed by an AI application.

Compute Administrator Role: Has the compute resources and data communications network (DCN) that support the management/control applications for the network in scope.

Network Administrator Role: Has the complete network in scope.

Controller Administrator Role: Has a single instance of a SDN controller in scope. The controller administrator can configure the controller and assign resources to clients (as described in section 6 Control Service and Various Deployments on page 14).

For this overview it is assumed that only one administrator role is active.

5.1 Platform creation (Compute administration)

- Assign compute resources.
  The model for compute resources is provided in TR-512.15
  The platform could be supported by (for example):

  - A dedicated bare metal (physical) computer.
  - A virtual machine (VM) running on a “local” server.
  - A VM running in a cloud computing environment.

  It is assumed that the initial compute resource has a default control port and “basic” bootstrap software that supports the ability to download software (to run on the platform).

  A dedicated physical compute platform can provide a high degree of isolation which may be desirable in some early deployment scenarios. However, the compute resources must be pre-engineered. A VM offers the possibility of allowing the compute resources assigned to a controller to be modified as the deployment evolves.

  It is desirable to be able to migrate a “running” controller between different compute platforms (e.g., from a dedicated bare metal (physical) computer to a VM).

- Install base controller software.
  This software creates a control construct that can be used to configure the rest of the controller.

- Create and configure the network administration and controller administration ports.
  This includes definition of the port address, protocols, and initial user credentials.

5.1.1 Platform modification or migration

The compute administration role is also responsible for:

- Modifying the resources (including compute, storage and the DCN) available to a running controller
- Deletion of a controller
- Migrating the controller to a different platform.
  It may be necessary to suspend controller operations during the migration, apart from this, migration should not be visible to the controller.

5.2 Assign network resources (network administration)

The network administrator identifies the transport resources that are intended to be within the scope of this controller and the way that these resources can be accessed. The resources identified include any links that are between a subnetwork in one server context and a subnetwork in another server context (i.e., in the scope of this controller) or between a subnetwork in the scope of this controller and a subnetwork that is in the scope of another controller.

Local resources are accessed directly by this controller. Any resources within the constraint domain that defines the boundary of the NE are in the scope of this server context. Note that this includes the resources that are used to support the control ports.

Resources provided via other controllers are accessed via a client context on a server controller. All of the resources in a (supporting) client context are in scope for the server context. The compute administrator should configure the DCN (if required) and provide the address and the initial credentials required to access these resources.

5.3 Configure server contexts (controller administrator)

- Configure server contexts:
  - Configure view translation which includes:
    View normalization i.e., mapping the semantics of the representation to the local model semantics and:
    Translating the name space (in the server controller client context) to the local controller (common) name space.
  - For server contexts that will access resources provided via other controllers: Create communications ports with user groups and user credentials (to access the supporting client contexts).
- Confirm resources:
  - Local resources: The resources retrieved from the (local) constraint domain should be refactored (i.e., mapped from the server specific representation to the common local representation) and mapped to the name space of the local controller and placed in the local context/RDB. These (retrieved) resources should be checked against the planned resources provisioned by the network administrator, any discrepancies should be

flagged and resolved.
In a higher-level controller, or in the initial configuration of a low-level controller, the local resources may only support the control communications network (e.g., Ethernet interfaces).
The controller administrator will configure and allocate these communications resources to support the administration ports of this controller, ports on the server contexts and ports on the client contexts. The controller administrator should use the parameters provided by the compute administrator.

  - The resources retrieved by a server context from a corresponding (supporting) client context should be refactored (i.e., mapped from the server specific representation to the common local representation) and name space of the local controller and placed in the local context/RDB. These (retrieved) resources should be checked against the planned resources provisioned by the network administrator, any discrepancies should be flagged and resolved.

- Confirm links: Between subnetworks in different server contexts in this controller or between subnetworks in the scope of this controller and subnetworks in the scope of another controller.
  - The links can be retrieved automatically by using the discovery process defined in [ITU-T G.7714] or by using a manual test procedure. These retrieved links should be checked against the links identified by the network administrator, any discrepancies should be flagged and resolved.

An example of a controller after the steps listed above have been completed is illustrated in Figure 5-1 below. Note that the links shown in the figure are between the resources in the server context (and not between the server contexts).

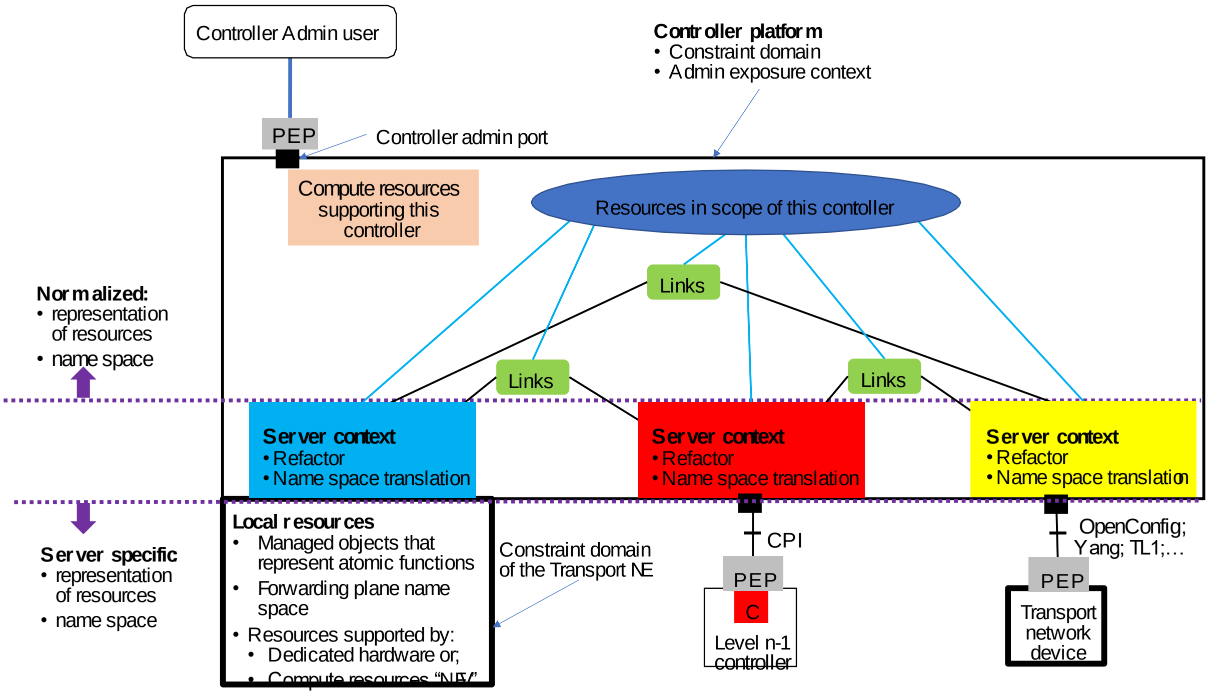


Figure 5-1 Example of a controller after initial configuration

On completion of the steps the controller does not support any clients, the controller can be configured to support clients as described in section 6.

6 Control Service and Various Deployments

In this section it is assumed that a controller has been created and configured as described in section 5 and that the DCN has been configured to support clients. This section provides an overview of the process for configuration of the controller to support clients.

Example deployment scenarios are described in section 6.1. The client contexts (CCs) to support these scenarios are described in section 6.2. In all cases the client (of a CC) is a single server context (or application). A client context may support more than one exposure context (EC), each EC has an independent API in a common port group.

The construction of a CC to support scenario 4 is described in section 6.3.

6.1 Example deployment scenarios

A SDN controller may be deployed in a number of different network contexts that provide different management control capabilities to a client. Some example deployment scenarios, from the simplest to most complex, are described below starting with the least complex. These examples are only intended to be used to check that the model is “fit for purpose”.

1) Controller in a NE with a <u>single client</u> in a <u>trusted domain</u>

- One client context (CC) with one exposure context (EC).
- Single server context with local resources.
- Link management is not supported.

This example shows the simplest (i.e., minimal) configuration of a controller.

2) Controller that consolidates multiple NEs with a <u>single client</u> in a <u>trusted domain</u>

- One client context with one or more ECs that provide different resource views and/or control capabilities to the same client.
  - Controller admin defines the EC and (client) translation function.
- Multiple server contexts.
- Management of links between the transport resources in the server contexts.

3) Controller with <u>more than one client</u> in a <u>trusted domain</u>

- More than one client context each with one or more EC
  - Controller admin defines the EC in the CC and the (client) translation function.
- Multiple server contexts
- Management of links between the transport resources in the server contexts.

4) Controller with <u>more than one client</u> in a <u>non-trusted domain with client admin control</u>

- More than one client context each with one or more EC

- Client controller admin:
    - Defines the EC in the CC and the (client) translation function.
    - Manages the client's users (i.e., configures the user credentials).
- Controller admin defines maximum "rate" of operations that the client can perform.
- Controller admin defines the maximum number of ECs that the Client admin can create.

6.2 Multiple views

Sketches of the client context for each of the scenarios described in section 6.1 are provided below. In these sketches:

- The constraint domain boundary coincides with an exposure context boundary.
- The control constructs are not shown.

For scenarios 2 to 4 more than one API is provided between the server context and the client context, these APIs are collected into a single port group.

For scenarios 3 and 4 multiple (independent) CCs exist, to avoid clutter on the figures only one CC is shown. The control constructs are shown in section 6.3 which provides an overview of the process for building a client context to support scenario 4.

6.2.1 Controller in a NE with a <u>single client</u> in a <u>trusted domain</u>

This is an example of a controller embedded in a Network Element. In this example the controller has a single server context (that contains the local resources), and a single client context with one user that has full access to the resources.

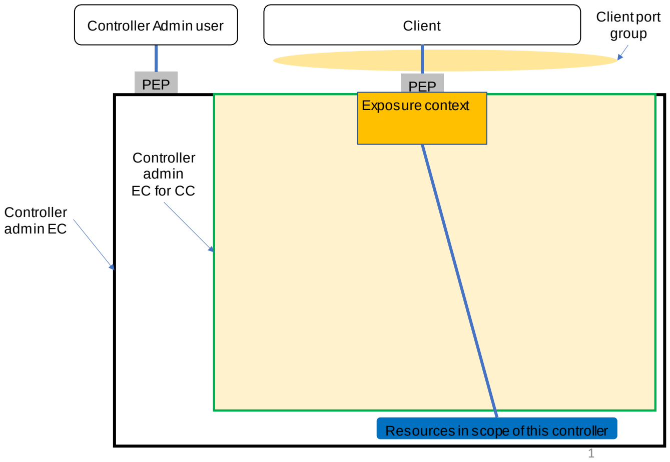


Figure 6-1 Controller in a NE with a single client in a trusted domain

The controller administrator manages the client’s users (i.e., configures the user credentials).

In this example the server context maps the local resource representation (e.g., OpenConfig) directly to the client name space and representation (e.g., TAPI) i.e., the client name space and representation is used for the internal (local) name space and representation. This is the simplest configuration and could be used for example in a controller embedded in a network element that only supports local resources.

6.2.2 Controller that consolidates multiple NEs with a single client in a trusted domain

This is an example of a controller used in a hierarchy of controllers, in a single administration, to provide scalability by consolidating the interfaces of multiple NEs or subordinate controllers into a single interface. One client with multiple users is supported.

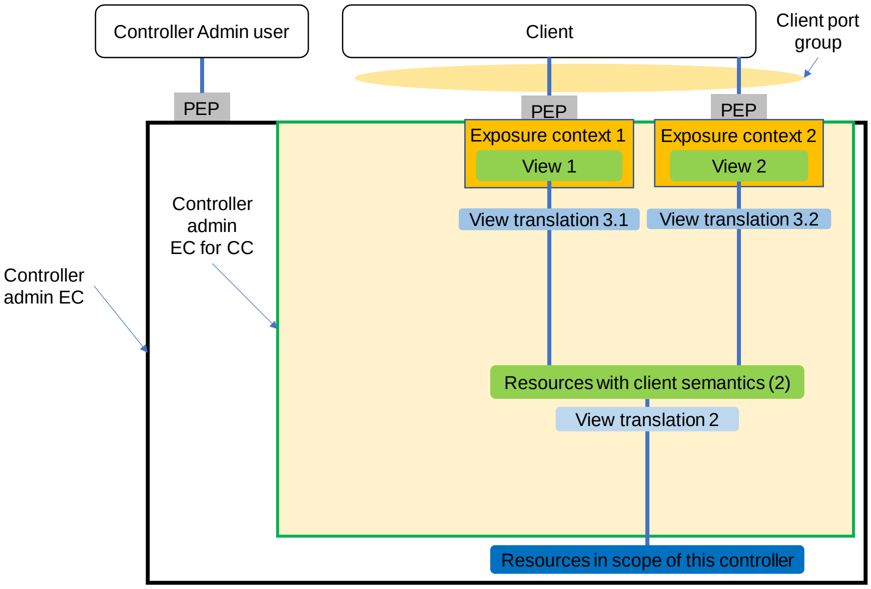


Figure 6-2 Controller that consolidates multiple NEs with a single client in a trusted domain.

The controller administrator manages the exposure contexts provided for the client. The controller administrator also manages the and client’s users (i.e., configures the user credentials).

View translation 2 is configured by the controller administrator. View translation 2 refactors (i.e., maps from the local representation to the client representation) and maps to the local name space of the client name space. This function will be null if the server contexts map the representation and name space of the resources directly to the representation and name space used for the client. In this case the client context and the controller use the same semantics and name space. This is possible since the controller only has a single client, and the controller and client are in the same trusted domain.

Resources with client semantics (2) provides the most detailed view available to the client and allows full control of the resources.

View translation 3.x is configured by the controller administrator. It provides a view that is appropriate for the target user. The resources in view 3.x may be:

- A subset of the resources in view (2).
- Access may be limited (e.g., some attributes may be set to “read only”)
- Some resources may be aggregates (e.g., a set of links and subnetworks may be represented as a single subnetwork).

6.2.3 Controller with more than one client in a trusted domain

This is an example of a controller used in a hierarchy of controllers, in a single administration, to provide scalability by consolidating the interfaces of multiple NEs or subordinate controllers into a single interface. Multiple clients with multiple users are supported. Each client has a dedicated client context. Only a single client (and the corresponding client context) is shown in Figure 6-3 below.

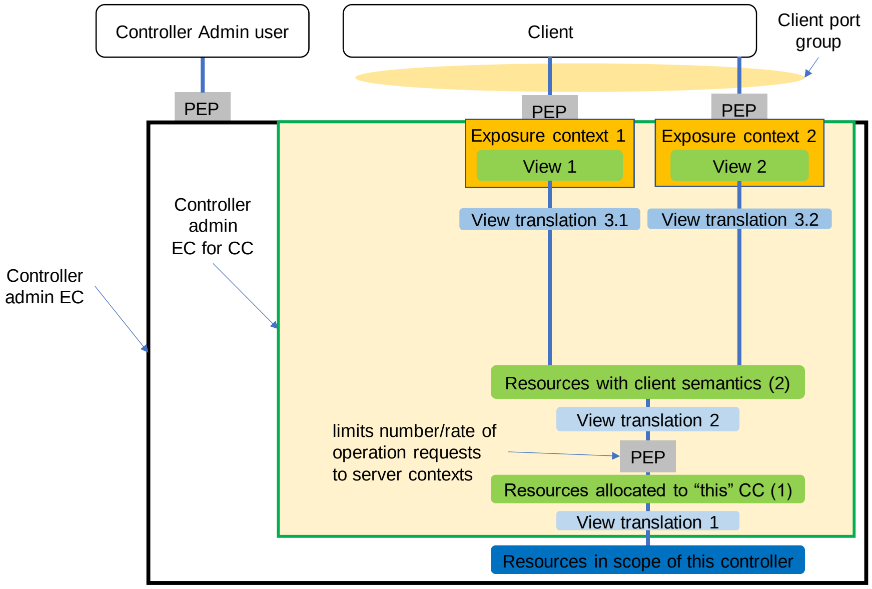


Figure 6-3 Controller with more than one client in a trusted domain

The controller administrator manages the exposure contexts provided for the client. The controller administrator also manages the client’s users (i.e., configures the user credentials).

View translation 1 is managed by the controller administrator. It provides a subset of the resources in the scope ot the controller to the client context and manages the availability status of the resources allocated to the CC. This enables resources to be shared by multiple clients.

Resources allocated to this CC (1) is a subset of the resources in the scope of the controller, if resources are shared between clients the availability status controlls the ability of the client to use the resources.

The PEP limits the number and rate of requests that are supported for the client. This is intended to prevent one client from overloading the compute resources of the controller which would impact the capabilities provided to other clients.

View translation 2 is managed by the controller administrator. View translation 2 refactors (i.e., maps from the local representation to the client representation) and maps to the local name space of the client name space (as described above). It may also:

- Limit access (e.g., some attributes may be set to “read only”)
- Aggregate some resources (e.g., a set of links and subnetworks may be represented as a single subnetwork).

Resources with client semantics (2) and view translation 3.x are as described above.

This configuration may be used, for example, in a network that supports both OTN and Ethernet services, but the client’s users can only access either OTN or Ethernet resources (but not both).

6.2.4 Controller with more than one client in a non-trusted domain with extended client control

This is an example of a controller used in a hierarchy of controllers, that consolidates the interfaces of multiple NEs or subordinate controllers into a single interface. Multiple clients with multiple users are supported. Each client has a dedicated client context. Only a single client (and the corresponding client context) is shown in Figure 6-4 below.

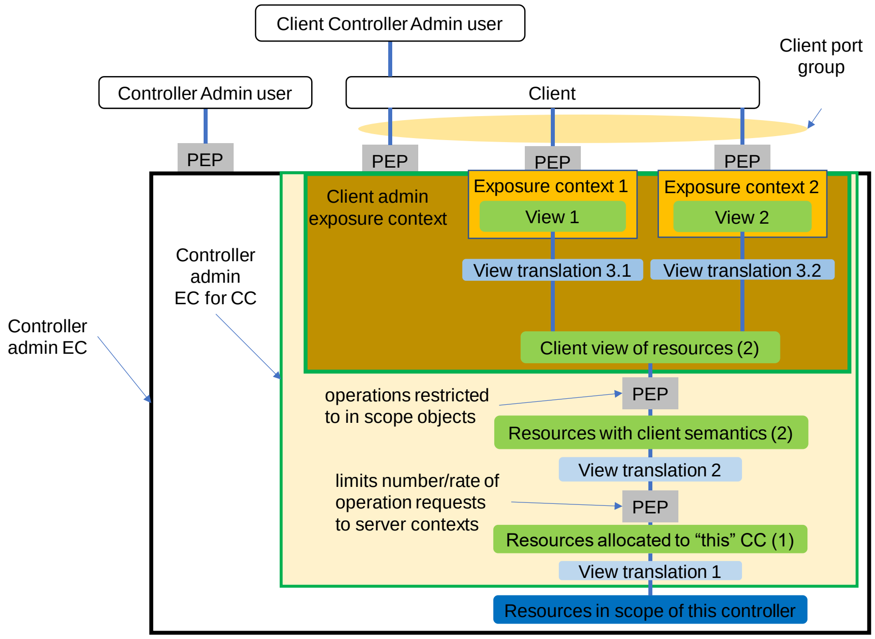

Client Controller Admin user
Client port group
Controller Admin user
Client
PEP
PEP
PEP
PEP
Client admin exposure context
Exposure context 1
View 1
Exposure context 2
View 2
Controller admin EC for CC
View translation 3.1
View translation 3.2
Controller admin EC
Client view of resources (2)
operations restricted to in scope objects
PEP
Resources with client semantics (2)
View translation 2
limits number/rate of operation requests to server contexts
PEP
Resources allocated to “this” CC (1)
View translation 1
Resources in scope of this controller

Figure 6-4 Controller with more than one client in a non-trusted domain

In this scenario the admin user in the client controller has the ability to control some aspects of the client context, using the client admin interface within the context of the client admin exposure context. The PEP on the client admin interface rejects any operations that are not permitted.

The controller administrator manages the client admin exposure context and defines the maximum number of exposure contexts that the client administrator can create.

The client controller administrator manages the exposure contexts (including view translation 3.x) and users (i.e., configures the user credentials). .

Before accepting any operations, the PEP checks that the resources are in scope for the CC.

View translation (1), Resources allocate to this controller (1), PEP, View translation (2) and Resources with client semantics (2) are as described above.

This configuration may be used, for example, in the case where a client wishes to directly manage its user views and users.

6.3 Building a client context

This section provides an overview of the process to construct a client context to support scenario 4.

The configuration is performed in two steps, first the controller administrator creates the client context and assigns resources then the client administrator configures the user views and credentials.

6.3.1 Controller administrator

The controller administrator:

1. Creates the constraint domain and exposure context for the client context then:
    - Assigns compute resources.
    - Installs the required software.
    - Configures view translation 1:
        - This provides a view of the subset of the resources in the scope of the controller that are being provided to the client in the semantics and name space of the controller.
    - Configures view translation 2: This translation may include:
        - Refactoring
        - Abstraction
        - Name space translation
2. Creates the exposure context and constraint domain for the client administrator then:
    - Assigns compute resources.
        - Client admin exposure context should “run” in container (e.g., a VM; discrete compute platform; …) to ensure that the client cannot exceed the allocated compute resources.
    - Installs the required software.
    - Creates a port and the user credentials for the client administrator.
3. Defines constraints for example:
    - Maximum number of exposure contexts that can be created in the client context.

- Complexity of the view translation e.g.,
    - abstraction and subsets – allowed.
    - view transformation not supported.
- Maximum number of users
- Maximum rate of requests/operations
    - A single request from a client may result in multiple requests to the servers. For example, a request for a connection across an abstract sub-network will:
        - Require path computation (using the cc compute resources):
        - Multiple connection requests to the server contexts

The client context after the steps above are completed is shown in Figure 6-5 below.

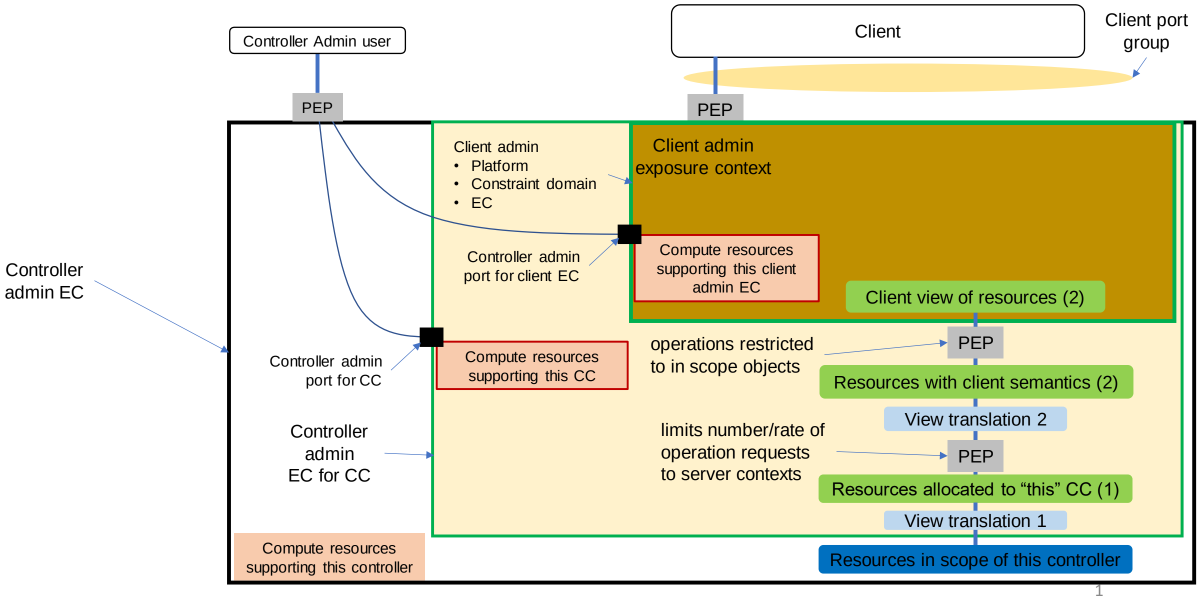

Controller Admin user
Client
Client port group
PEP
PEP
Controller admin EC
Client admin
• Platform
• Constraint domain
• EC
Client admin exposure context
Controller admin port for client EC
Compute resources supporting this client admin EC
Client view of resources (2)
Controller admin port for CC
Compute resources supporting this CC
operations restricted to in scope objects
PEP
Resources with client semantics (2)
View translation 2
Controller admin EC for CC
limits number/rate of operation requests to server contexts
PEP
Resources allocated to "this" CC (1)
View translation 1
Compute resources supporting this controller
Resources in scope of this controller
1

Figure 6-5 Configuration of a client context – step 1

6.3.2 Client controller administrator

Within the constraints of the client admin EC, defined by the controller administrator, the client controller administrator:

- Configures view translation 3.x.
- Creates exposure contexts for user groups.
- Creates user credentials (to access the exposure contexts).

The client context after these steps have been completed is shown in Figure 6-6 below.

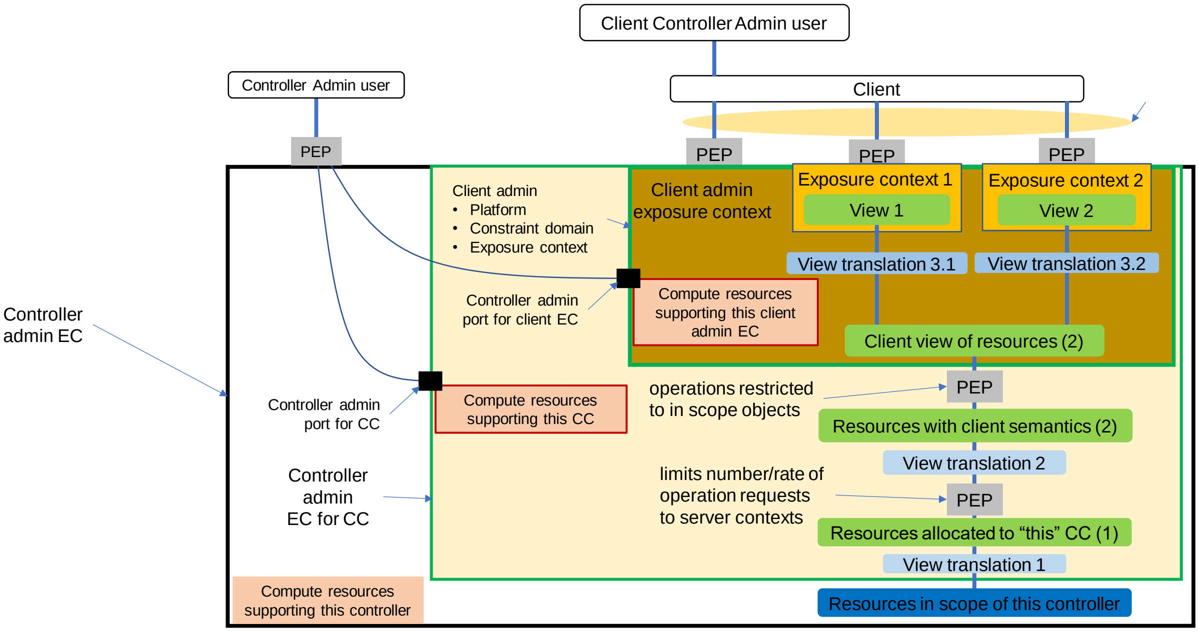


Figure 6-6 Configuration of a client context – step 2

7 Security Considerations

7.1 Deployment assumptions

- A client context only exists in the context of a “closed” environment based on a trusted commercial relationship.
- The Ports on the controller are in a secure environment.
  - e.g., behind a firewall
    - In the case of a controller embedded in a NE (shown in scenario 1), if the NE is an IP device the control traffic and user traffic may share the same port. This creates a potential point of vulnerability.
      In a “traditional” transport NE the control port and traffic ports are fully isolated
  - The client ports should be isolated from the controller admin port e.g., in separate VPNs
- Communications between the server context port and the client context port are secured by the protocol used on the link
  e.g., the protocol:
  - includes non-repudiation.
  - protects against man-in-the-middle attacks.
- A server context does not accept (management) connection requests.
  - The server context only initiates the set-up of (management) connections/sessions or streams.
    - The controller admin provides the credentials and address of the server (client context)
- Protocol includes “keep alive”.
  - If a connection drops the server context reinitiates the connection

7.2 Access control for users on the EC port

Exposure contexts (and access to them) are established in the context of a trusted commercial relationship.

- Policy Enforcement Point (PEP) on each EC port
  - Validates user credentials.
  - Policy Decision Point (PDP) may be local or remote.

User access to resources is controlled (constrained) by the view (EC) that is made available to that user.

An Exposure Context may be accessed using a Controlled Access Session (that is typically long lived). [PSBAC] describes patterns for access control sessions. Pattern based access control provides a similar function to Role-Based Attribute Control (RBAC).

Section 2.8 of [PSBAC] describes consequences of using a controlled access session.

> "This pattern has the following advantages:
>
> - We can give to each context only the needed rights according to its function and we can invoke in a session only those contexts that are needed for a given task.
> - We can exclude combinations of contexts that might result in possible access violations or conflicts of interest.
> - Once a subject starts a session it doesn't have to be reauthenticated. Its status is kept by the session.
>
> A possible disadvantage:
>
> - If we need to apply fine-grained access it might be inefficient to open many sessions to perform complex activities."

This is not a concern for Telecoms applications with a limited number of exposure contexts and user groups.

Access control may also be used by or implemented within call control in [ITU-T G.7701]. Call control protocols such as OIF UNI/ENNI and [ITU-T Q.2931] provide features such as peer authentication and closed user groups.

8 Timeline Examples

See section 3 Overview and Context on page 6 and section 4 Formation of a Controller on page 8 for background on lifecycle.

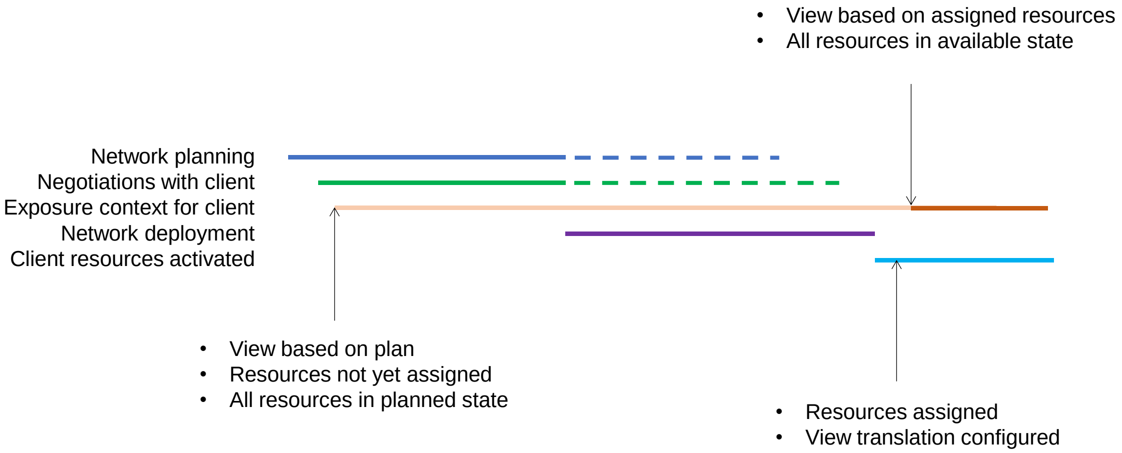

View based on assigned resources
All resources in available state
Network planning
Negotiations with client
Exposure context for client
Network deployment
Client resources activated
View based on plan
Resources not yet assigned
All resources in planned state
Resources assigned
View translation configured

Figure 8-1 Example time-lines – new client context

1) Market research and service catalog creation

- Initially based on a tentative plan – few (if any) resources deployed.
- Open environment – should run in an isolated “container” to limit damage from DoS attacks.

2) Negotiation with (potential) clients

- Based on firm plan: Service offerings defined: Some resources may have been deployed
- Open environment – should run in an isolated “container” to limit damage from DoS attacks.

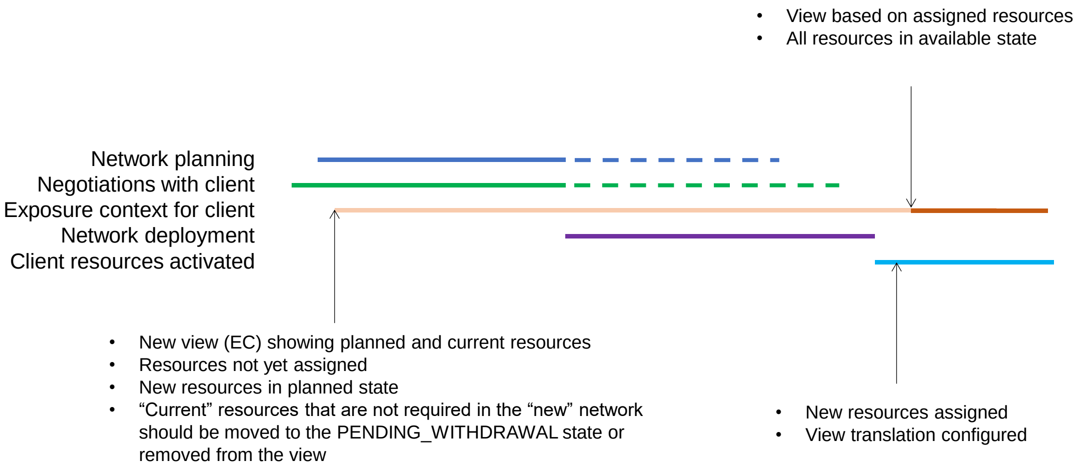
3) Select client and negotiate contract

- Establish a trusted commercial relationship
- Create a secure (closed) environment
- Controller and client context created

- May be created on in a "test" environment
- Client exposure context showing the resources available to the client
    - Mix of planned and deployed resources all with an assignment state of PLANNED in the client EC

4) Activation of client network

- Resources deployed
    - Possibly migrate the controller to a different platform
    - Migrate the "planning" EC to an operational EC – updated assignment state of resources



- View based on assigned resources
- All resources in available state

- New view (EC) showing planned and current resources
- Resources not yet assigned
- New resources in planned state
- "Current" resources that are not required in the "new" network should be moved to the PENDING_WITHDRAWAL state or removed from the view

- New resources assigned
- View translation configured

Figure 8-2 Example time-lines – modify existing client context

1) Create "planning" EC for the redesigned network

- Controller administrator adds "planned" resources to the client context
    - Additional resources are shown with an assignment state of PLANNED
        - These may be from a "planned" server context or deployed resources

- The “planning” view includes read only proxy for all the resources in the current network
    - i.e., the current network cannot be modified in the “planning” EC
    - The planned resources are not visible in the “active” view
- The client tags resources that are not required in the redesigned network with an assignment state of PENDING_WITHDRAWAL

2) Activation of new client network

- Plan is finalized and committed
    - Transition to the “new” network will be implemented at some defined time in the future
    - Migration from the current to new network is planned
        - This may be a multi-step process to minimize disruption during network migration
        - May require the temporary assignment of resources to minimize disruption during migration
- Resources deployed
- Migrate the resources in the “planning” EC to the “current” EC
    - Update resource assignment state
    - May involve several intermediate steps (as defined in the migration plan)

End of Document