Network Working Group
Internet-Draft
Intended status: Informational
Expires: ~~May 25,~~ **June 4,** 2020

M. Boucadair
C. Jacquenet
Orange
D. Zhang
Huawei Technologies
P. Georgatsos
CERTH

~~November 22,~~
December 2, 2019

Connectivity Provisioning Negotiation Protocol (CPNP)

~~draft-boucadair-connectivity-provisioning-protocol-17~~
draft-boucadair-connectivity-provisioning-protocol-18

Abstract

This document specifies the Connectivity Provisioning Negotiation Protocol (CPNP) which is designed for dynamic negotiation of service parameters.

CPNP is a generic protocol that can be used for various negotiation purposes that include (but are not necessarily limited to) connectivity provisioning services, storage facilities, Content Delivery Networks footprint, etc. The protocol can be extended with new Information Elements.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on ~~May 25,~~ **June 4,** 2020.

Table of Contents

1. Introduction

This document defines the Connectivity Provisioning Negotiation Protocol (CPNP) that is meant to dynamically exchange and negotiate connectivity provisioning parameters, and other service-specific parameters, between a Customer and a Provider. CPNP is a tool that introduces automation in the service negotiation and activation procedures, thus fostering the overall service provisioning process. CPNP can be seen as a component of the dynamic negotiation meta-domain described in Section 3.4 of [RFC7149].

CPNP is a generic protocol that can be used for other negotiation purposes than connectivity provisioning. For example, CPNP can be used to request extra storage resources, to extend the footprint of a CDN (Content Delivery Networks), to enable additional features from a cloud Provider, etc. CPNP can be extended with new Information Elements (IEs).

[RFC7297] describes a Connectivity Provisioning Profile (CPP) template to capture connectivity requirements to be met by a transport infrastructure for the delivery of various services such as Voice over IP (VoIP), IPTV, and Virtual Private Network (VPN) services [RFC4026]. The CPP document defines the set of IP transfer parameters that reflect the guarantees that can be provided by the underlying transport network together with reachability scope and capacity needs. CPNP uses the CPP template to encode connectivity provisioning clauses ~~during~~ **under** negotiation. **The agreed CPP will be then passed to other functional elements that are responsible for the actual service activation and provisioning. For example, NETCONF [RFC6241] or RESTCONF [RFC8040] can be used to activate adequate network features that are required to deliver the agreed service. How the outcome of CPNP negotiation is translated into service and network provisioning actions is out of scope of this document.**

As a reminder, several proposals have been made in the past by the (research) community (e.g., COPS-SLS, Service Negotiation Protocol ~~(SrNP)~~

~~,~~
(SrNP), Dynamic Service Negotiation Protocol (DSNP), Resource Negotiation and Pricing Protocol (RNAP), Service Negotiation and Acquisition Protocol ~~(SNAP)~~

~~).~~ **(SNAP)).** CPNP leverages the experience of the authors with SrNP by separating the negotiation primitives from the service under negotiation. Moreover, careful examination of the other proposals revealed certain deficiencies that were easier to address through the creation of a new protocol rather than modifying existing protocols. For example:

- COPS-SLS relies upon COPS-PR [RFC3084], which is an Historic RFC.
- DSNP is tightly designed with one specific service in mind (QoS) and does not make any distinction between a quotation phase and

the actual service ordering phase.

One of the primary motivations of this document is to provide a permanent reference to exemplify how service negotiation can be automated.

This document is organized as follows:

- Section 3 defines the functional elements involved in CPNP exchanges.
- Section 4 introduces several order processing models and precises those that are targeted by CPNP.
- Section 5 enumerates a non-exhaustive list of use cases that could benefit from CPNP.

- Section 5 discusses CPNP deployment models.
- Section 7 presents the CPNP negotiation model.
- Section 8 provides an overview of the protocol.
- Section 9 specifies the CPNP objects.
- Section 10 describes the CPNP message validation procedure.
- Section 11 specifies the behavior of the involved CPNP functional elements.
- Section 12 discusses relevant operational guidelines.
- Section 13 discusses protocol security aspects.

Implementation details are out of scope. An example of required modules and interfaces to implement this specification is sketched in Section 4 of [AGAVE]. This specification builds on that effort.

2. Terminology

This document makes use of the following terms:

Customer: Is a business role which denotes an entity that is involved in the definition and the possible negotiation of a contract, including a Connectivity Provisioning Agreement, with a Provider. A connectivity provisioning contract is captured in a dedicated CPP template-based document, which specifies (among other information): the sites to be connected, border nodes, outsourced operations (e.g., routing, force via points).

The right to invoke the subscribed service may be delegated by the Customer to third-party End Users, or brokering services.

A Customer can be a Service Provider, an application owner, an enterprise, a user, etc.

Network Provider (or Provider): Owns and administers one or many transport domain(s) (typically Autonomous System (AS)) composed of IP switching and transmission resources (e.g., routing, switching, forwarding, etc.). Network Providers are responsible for ensuring connectivity services (e.g., offering global or restricted reachability at specific rates). Offered connectivity services may not necessarily be restricted to IP.

The policies to be enforced by the connectivity service delivery components can be derived from the technology-specific clauses that might be included in contracts agreed with the Customers. If no such clauses are included in the agreement, the mapping between the connectivity requirements and the underlying technology-specific policies to be enforced is deployment-specific.

Quotation Order: Denotes a request made by the Customer to the Provider that includes a set of requirements. The Customer may express its service-specific requirements by assigning (fixed or loosely defined) values to the information items included in the commonly understood template (e.g., CPP template) describing the offered service. These requirements constitute the parameters to be mutually agreed upon.

Offer: Refers to a response made by the Provider to a Customer 's quotation order as to the extent at which the Provider may satisfy the order at the time of its receipt. Offers reflect the capability of the Provider in accommodating received Customer orders beyond monolithic 'yes/no' answers.

An offer may fully or partially meet the requirements of the corresponding order. In the latter case, it may include alternative suggestions which the Customer may take into account by issuing a new order.

Agreement: Refers to an order placed by the Customer and accepted by the Provider. It signals the successful conclusion of a negotiation cycle.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119][RFC8174] when, and only when, they appear in all capitals, as shown here.

3. CPNP Functional Elements

The following functional elements are defined:

CPNP client (or client): Denotes a software instance that sends CPNP requests and receives CPNP responses. The current operations that can be performed by a CPNP client are listed below:

1. Create a quotation order (Section 9.2.1).

2. Cancel an ongoing quotation order under negotiation (Section 9.2.7).

3. Accept an offer made by a server (Section 9.2.4).

4. Withdraw an agreement (Section 9.2.8).

5. Update an agreement (Section 9.2.9).

CPNP server (or server): Denotes a software instance that receives CPNP requests and sends back CPNP responses accordingly. The CPNP server is responsible for the following operations:

1. Process a quotation order (Section 9.2.2).

2. Make an offer (Section 9.2.3).

3. Cancel an ongoing quotation order (Section 11.2.3).

4. Process an order withdrawal (Section 11.2.3).

4. Order Processing Models

For preparing their service orders, the Customers may need to be aware of the offered services. The Providers therefore should first proceed with the announcement of the services that they can provide. The service announcement process may take place at designated global or Provider-specific service markets, or through explicit interactions with the Providers. The details of this process are outside the scope of a negotiation protocol.

With or without such service announcement mechanisms in place, the following order processing models can be distinguished:

Frozen model:

> The Customer cannot actually negotiate the parameters of the service(s) offered by a Provider. After consulting the Provider's service portfolio, the Customer selects the service offer he/she wants to subscribe and places an order to the Provider. Order handling is quite simple on the Provider side because the service is not customized as per Customer's requirements, but rather pre-designed to target a group of customers having similar requirements (i.e., these customers share the same Customer Provisioning Profile). **This mode can be implemented using existing tools such as [RFC8309].**

Negotiation-based model:

> Unlike the frozen model, the Customer documents his/her requirements in a request for a quotation, which is then sent to one or several Providers. Solicited Providers check whether they can address these requirements or not, and get back to the Customer accordingly, possibly with an offer that may not exactly match customer's requirements (e.g., a 100 Mbps connection cannot be provisioned given the amount of available resources, but an 80 Mbps connection can be provided). A negotiation between the Customer and the Provider(s) then follows to the end of reaching an ~~agreement.~~ **agreement (or not).**

Both frozen and negotiation-based models require the existence of appropriate service templates like a CPP template and their instantiation for expressing specific offerings from Providers and service requirements from Customers, respectively. CPNP can be used in either model for automating the required Customer-Provider interactions. Since the frozen model can be seen as a special case of the negotiation-based model, not only 'yes/no' answers but also counter offers may be issued by the Provider in response to Customer orders, this document focuses on the negotiation-based model.

Order processing management on the Network Provider's side is usually connected with the following functional blocks:

- Network Provisioning (including Order Activation, Network Planning, etc.)
- Authentication, Authorization and Accounting (AAA)
- Network and service management (performance verification, complaint analysis, etc.)
- Sales-related functional blocks (e.g., billing, invoice validation)
- Network Impact Analysis

CPNP does not assume any specific knowledge about these functional blocks, drawing an explicit line between protocol operation and the logic for handling connectivity provisioning requests. Evidently order handling logic is subject to the information manipulated by these blocks. For example, the resources that can be allocated to accommodate Customer's requirements may depend on network availability estimates as calculated by the planning functions and related policies as well as on the number of orders to be processed simultaneously over a given period of time.

This document does not elaborate on how Customers are identified and subsequently managed by the Provider's Information System.

5. Sample Use Cases

A non-exhaustive list of CPNP use cases is provided below:

1. [RFC4176] introduces the L3VPN Service Order Management functional block which is responsible for managing the requests initiated by the Customers and tracks the status of the completion of the related operations. CPNP can be used between the Customer and the Provider to negotiate L3VPN service parameters.

   A CPNP server could therefore be part of the L3VPN Service Order Management functional block discussed in [RFC4176]. A YANG data model for L3VPN service delivery is defined in [RFC8299].

2. CPNP can be used between two adjacent domains to deliver IP interconnection services (e.g., enable, update, disconnect). For example, two Autonomous Systems (ASes) can be connected via several interconnection points. CPNP can be used between these ASes to upgrade existing links, request additional resources, provision a new interconnection point, etc.

   See, for example, the framework documented in [ETICS].

3. An integrated Provider can use CPNP to rationalize connectivity provisioning needs related to its service portfolio. A CPNP server function is used by network operations teams. A CPNP interface to invoke CPNP negotiation cycles is exposed to service management teams.

4. Service Providers can use CPNP to initiate connectivity provisioning requests towards a number of Network Providers so that to optimize the cost of delivering their services. Although multiple CPNP ordering cycles can be initiated by a Service Provider towards multiple Network Providers, a subset of these orders may actually be put into effect.

   For example, a cloud Service Provider can use CPNP to request more resources from Network Providers.

5. CPNP can also be used in the context of network slicing ([I-D.geng-netslices-architecture]) to request for network resources together with a set of requirements that need to be satisfied by the Provider. Such requirements are not restricted to basic IP forwarding capabilities, but may also include a characterization of a set of service functions that may be invoked.

6. CPNP can be used in Machine-to-Machine (M2M) environments to dynamically subscribe to M2M services (e.g., access to data retrieved by a set of sensors, extend sensor coverage, etc.).

   Also, Internet of Things (IoT, [RFC6574]) domains may rely on CPNP to enable dynamic provisioning of data produced by involved objects, according to their specific policies, to various external stakeholders such as data analytics and business intelligence companies. Direct CPNP-based interactions between IoT domains and interested parties enable open access to diverse sets of data across the Internet, e.g., from multiple types of sensors, user groups and/or geographical areas.

7. CPNP can be used in the context of I2NSF ([RFC8329]) to capture the customer-driven policies to be enforced by a set of Network Security Functions.

8. A Provider offering cloud services can expose a CPNP interface to allow Customers to dynamically negotiate related service features such as additional storage, processing and networking resources, enhanced security filters, etc.

9. In the inter-cloud context (also called cloud of clouds or cloud federation), CPNP can be used to reserve external computing and networking resources in other cloud environments.

10. CDN Providers can use CPNP to extend their footprint by interconnecting their CDN infrastructure [RFC6770] (see Figure 1).

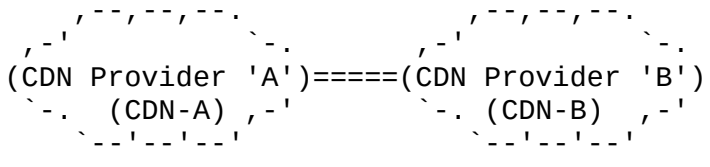


Figure 1: CDN Interconnection

11. Mapping Service Providers (MSPs, [RFC7215]) can use CPNP to enrich their mapping database by interconnecting their mapping system (see Figure 2). This interconnection allows to relax the constraints on PxTR in favour of native LISP forwarding [RFC6830]. Also, it allows to prevent fragmented LISP mapping database. A framework is described in [I-D.boucadair-lisp-idr-ms-discovery].

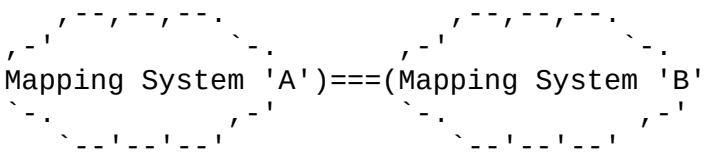


Figure 2: LISP Mapping System Interconnect

CPNP may also be used between SDN controllers in contexts where Cooperating Layered Architecture for Software-Defined Networking (CLAS) is enabled [RFC8597].

6. CPNP Deployment Models

Several CPNP deployment models can be envisaged. Two examples are

listed below:

- The Customer deploys a CPNP client while one or several CPNP servers are deployed by the Provider.
- The Customer does not enable any CPNP client. The Provider maintains a Customer Order Management portal. The Customer can initiate connectivity provisioning quotation orders via the portal; appropriate CPNP messages are then generated and sent to the relevant CPNP server. In this model, both the CPNP client and CPNP server are under the responsibility of the same administrative entity (i.e., Network Provider).

Once the negotiation of connectivity provisioning parameters is successfully concluded that is, an order has been placed by the Customer, the actual network provisioning operations are initiated. The specification of related dynamic resource allocation and policy enforcement schemes, as well as how CPNP servers interact with the network provisioning functional blocks at Provider sides are out of the scope of this document.

This document does not make any assumption about the CPNP deployment model either.

7. CPNP Negotiation Model

CPNP runs between a Customer and a Provider carrying service orders from the Customer and respective responses from the Provider to the end of reaching a connectivity service provisioning agreement. As the services offered by the Provider are well-described, by means of the CPP template, the negotiation process is essentially a value-settlement process, where an agreement is pursued on the values of the commonly understood information items (service parameters) included in the service description template.

The protocol is transparent to the content that it carries and to the negotiation logic, at Customer and Provider sides, that manipulates the content.

The protocol aims at facilitating the execution of the negotiation logic by providing the required generic communication primitives.

Since negotiations are initiated and primarily driven by the Customer's negotiation logic, it is reasonable to assume that the Customer can only call for an agreement. An implicit approach is adopted for not overloading the protocol with additional messages. In particular, the acceptance of an offer made by the Provider signals a call for agreement from the Customer. Note that it is almost certain the Provider to accept this call since it refers to an offer that itself made. Of course, at any point the Provider or the Customer may quit the negotiations, each on its own grounds.

Based on the above, CPNP adopts a Quotation Order/Offer/Answer model, which proceeds through the following basic steps:

1. The client specifies its service requirements via a Provision Quotation Order (PQO). The order may include fixed or loosely defined values in the clauses describing service provisioning characteristics.

2. The server declines the PQO, or makes an offer to address the requirements of the PQO, or which may suggests a counter-

proposals that partially addresses the requirements of the PQO for specific requirements that cannot be accommodated.

3. The client either accepts or declines the offer. Accepting the offer implies a call for agreement.

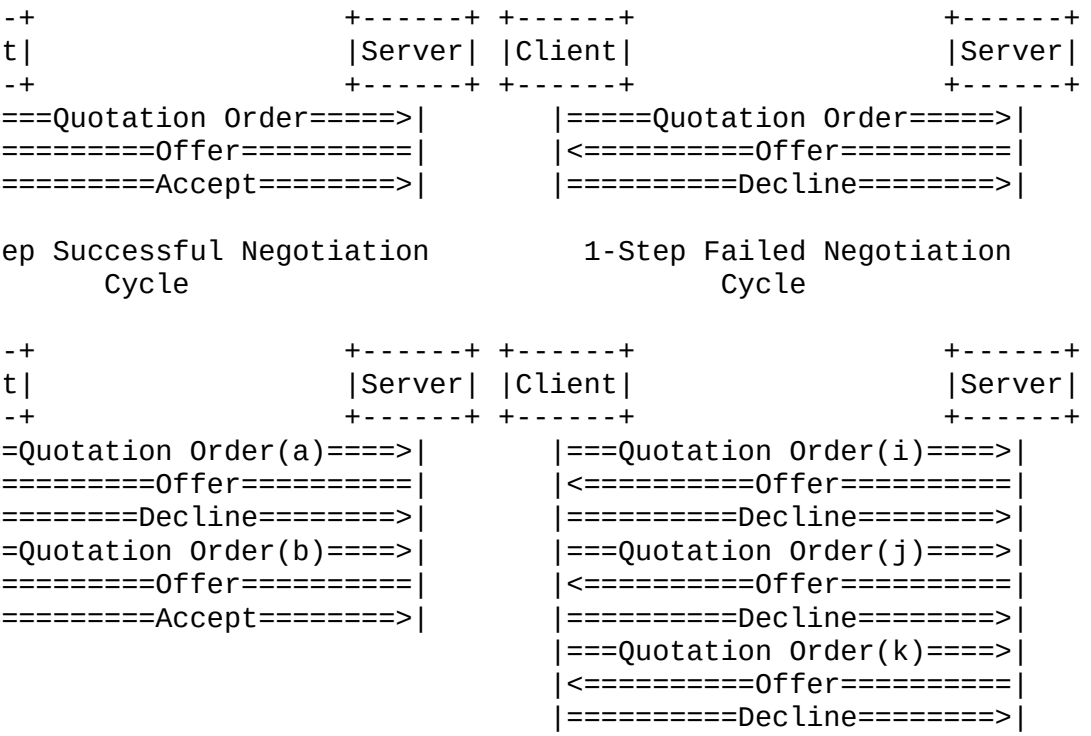
Multiple instances of CPNP may run at Customer or Provider domains. A CPNP client may be engaged simultaneously in multiple negotiations with the same or different CPNP servers (parallel negotiations, see Section 8.9) and a CPNP server may need to negotiate with other Provider(s) as part of negotiations with a CPNP client (cascaded negotiations, see Section 8.8).

CPNP relies on various timers to achieve its operations. Two types of timers are defined: those that are specific to CPNP message transmission and those that are specific to the negotiation logic. The latter are used to guide the negotiation logic at both CPNP client and CPNP server sides, particularly in cases where the CPNP client is involved in parallel negotiations with several CPNP servers or in cases where the CPNP server is, in its turn, involved in negotiations with other Providers for processing a given quotation order. Related to the above, CPNP allows the CPNP server to request for more time. This request may be accepted or rejected by the CPNP client.

Providers may need to publish available services to the Customers (see Section 4). CPNP may optionally support this functionality. Dedicated templates can be defined for the purpose of service announcements, which will be used by the CPNP clients to initiate their CPNP negotiation cycles.

For simplicity, a single Offer/Answer stage is assumed within one a CPNP negotiation cycle. Nevertheless, as stated before, multiple CPNP negotiation cycles can be undertaken by a CPNP client (see Figure 3).

The model is flexible as it can accommodate changing conditions over time (e.g., introduction of an additional VPN site).



```
                                        |===Quotation Order(l)====>|
                                        |<==Fail to make an offer==|

      N-Step Negotiation Cycle:          N-Step Negotiation Cycle:
        Successful Negotiation               Failed Negotiation
```

Figure 3: Overall Negotiation Process

This version of the protocol does not support means for a client to retrieve a list of active/agreed offers.

8. Protocol Overview

8.1. Client/Server Communication

CPNP is a client/server protocol can run over any transport protocol with UDP being the default transport mode secured with Datagram Transport Layer Security (DTLS) [RFC6347]. No permanent CPNP transport session needs to be maintained between the client and the server.

The CPNP client can be configured with the CPNP server(s) (typically, an IP address together with a port number) using manual or dynamic configuration means. For example, a Provider advertises the port number (CPNP_PORT) it uses to bind the CPNP service (e.g., using SRV [RFC2782]).

The client sends CPNP messages to CPNP_PORT. The same port number used as the source port number of a CPNP request sent to the server MUST be used by the server to reply to that request.

CPNP is independent of the IP address family.

CPNP retransmission is discussed in Section ~~11.4.~~ **11.4 for unreliable transports.**

8.2. Policy Configuration on the CPNP Server

As an input to its decision-making process, the CPNP server may be connected to various external modules such as: Customer Profiles, Network Topology, Network Resource Management, Orders Repository, AAA and Network Provisioning Manager (an example is shown in Figure 4).

These external modules provide inputs to the CPNP server, so that it can:

- Check whether a customer is entitled to initiate a provisioning quotation request.
- Check whether a customer is entitled to cancel an on-going order.
- Check whether administrative data (e.g., billing-related information) have been verified before starting handling the request.
- Check whether network capacity is available or additional capacity is required.
- Receive guidelines from network design and sales blocks (e.g., pricing, network usage levels, threshold on number of CPP templates that can be processed over a given period of time as a

function of the nature of the service to be delivered, etc.).

o Transfer completed orders to network provisioning blocks. For example, the outcome of CPNP may be passed to modules such as Application-Based Network Operations (ABNO) [RFC7491] or network controllers. **These controllers will use protocols such as NETCONF [RFC6241] to interact with the appropriate network nodes and functions for the sake of proper service activation and delivery.**

The above list of CPNP server operations is not exhaustive.

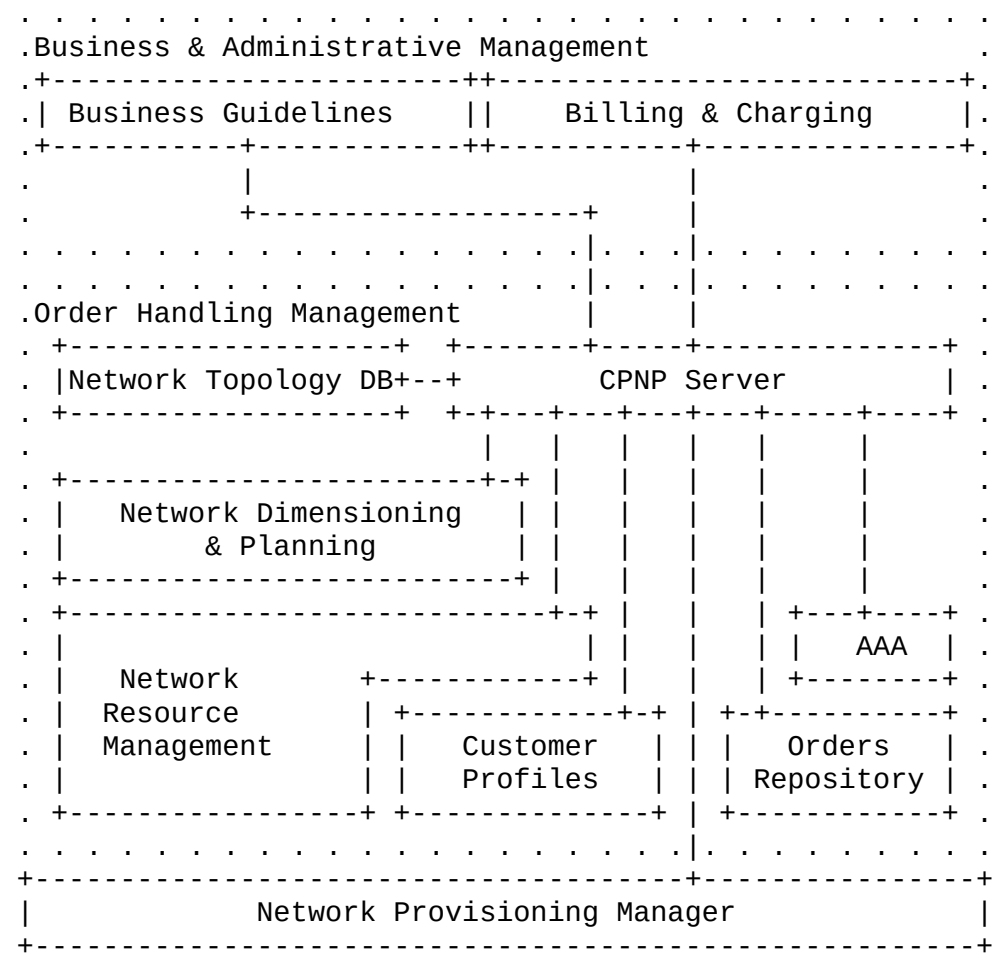


Figure 4: Order Handling Management Functional Block

The following order handling modes can be also configured on the server:

1. Fully automated mode: This mode does not require any action from the administrator when receiving a request for a service. The server can execute its decision-making process related to the orders received and generate corresponding offers.

2. Administrative validation checking: Some or all of the server's operations are subject to administrative validation procedures. This mode requires an action from the administrator for every request received. The CPNP methods which can be automatically handled by the server or they are subject to one or several validation administrative checks can be configured on the server.

8.3. CPNP Session Entries

A CPNP session entry is denoted by a 5-uplet defined as follows:

- Transport session (typically, IP address of the client, client's port number, IP address of the server, and server's port number).

- Incremented Sequence Number (Section 11.3)

- Customer Agreement Identifier: This is a unique identifier assigned to the order under negotiation by the client (Section 9.1.1). This identifier is also used to identify the agreement that will result from a successful negotiation.

- Provider Agreement Identifier: This is a unique identifier assigned to the order under negotiation by the server (Section 9.1.2). This identifier is also used to identify the agreement that will result from a successful negotiation.

- Transaction-ID (Section 9.1.3).

8.4. CPNP Transaction

A CPNP transaction occurs between a client and a server for pursuing, modifying, withdrawing a service agreement, and comprises all CPNP messages exchanged between the client and the server, from the first request sent by the client to the final response sent by the server. A CPNP transaction is bound to a CPNP session (Section 8.3).

Because multiple CPNP transactions can be maintained by the CPNP client, the client must assign an identifier to uniquely identify a given transaction. This identifier is denoted as Transaction-ID.

The Transaction-ID must be randomly assigned by the CPNP client, according to the best current practice for generating random numbers

[RFC4086] that cannot be guessed easily. Transaction-ID is used for validating CPNP responses received by the client.

In the context of a transaction, the client needs to randomly select a sequence number and assign it in the first CPNP message to send. This number is then incremented for each request message is subsequently sent within the on-going CPNP transaction (see Section 11.3).

8.5. CPNP Timers

CPNP adopts a simple retransmission procedure which relies on a retransmission timer denoted as RETRANS_TIMER and maximum retry threshold. The use of RETRANS_TIMER and a maximum retry threshold are described in Section 11.

The response timer (RESPONSE_TIMER) is set by the client to denote the time, in seconds, the client will wait for receiving a response from the server to a provisioning quotation order request (see Section 9.1.6). If the timer expires, the respective quotation order is cancelled by the client and a CANCEL message is generated accordingly.

An offer expiration timer (EXPIRE_TIMER) is set by the server to represent the time, in minutes, after which an offer made by the server will be invalid (see Section 9.1.8).

8.6. CPNP Operations

CPNP operations are listed below. They may be augmented, depending on the nature of some transactions or because of security considerations that may necessitate a distinct CPNP client/server authentication phase before negotiation begins.

- QUOTATION (Section 9.2.1):

  This operation is used by the client to initiate a provisioning quotation order. Upon receipt of a QUOTATION request, the server may respond with a PROCESSING, OFFER or a FAIL message. A QUOTATION-initiated transaction can be terminated by a FAIL message.

- PROCESSING (Section 9.2.2):

  This operation is used to inform the remote party that the message (the order quotation or the offer) sent was received and it is processed. This message can also be issued by the server to request more time, in which case the client may reply with an ACK or FAIL message depending on whether more time can or cannot be granted.

- OFFER (Section 9.2.3):

  This operation is used by the server to inform the client about an offer that can best accommodate the requirements indicated in the previously received QUOTATION message.

- ACCEPT (Section 9.2.4):

  This operation is used by the client to confirm the acceptance of an offer made by the server. This message implies a call for agreement. An agreement is reached when an ACK is subsequently received from the server, which is likely to happen; it is rather unlikely the server to reject an offer that it has already made.

- DECLINE (Section 9.2.5):

  This operation is used by the client to reject an offer made by the server. The on-going transaction may not be terminated immediately, e.g., the server/client may issue another offer/ order.

- ACK (Section 9.2.6):

  This operation is used by the server to acknowledge the receipt of an ACCEPT or WITHDRAW message, or by the client to confirm the time extension requested by the server for processing the last received quotation order.

- CANCEL (Section 9.2.7):

  This operation is used by the client to cancel (quit) the on-going transaction.

- WITHDRAW (Section 9.2.8):

  This operation is used by the client to withdraw an agreement.

- UPDATE (Section 9.2.9):

This operation is used by the client to update an existing agreement. For example, this method can be invoked to add a new site. This method will trigger a new negotiation cycle.

- FAIL (Section 9.2.10):

  This operation is used by the server to indicate that it cannot accommodate the requirements documented in the PQO conveyed in the QUOTATION message or to inform the client about an error encountered when processing the received message. In either case, the message implies that the server is unable to make offers and as such it terminates the on-going transaction.

  This message is also used by the client to reject a time extension request received from the server (in a PROCESSING message). The message includes a status code for providing explanatory information.

The above CPNP primitives are service-independent. CPNP messages may transparently carry service-specific objects which are handled by the negotiation logic at either side.

The document specifies the service objects that are required for connectivity provisioning negotiation (see Section 8.7). Additional service-specific objects to be carried in the CPNP messages can be defined in the future for accommodating alternative deployment or other service provisioning needs.

8.7. Connectivity Provisioning Documents

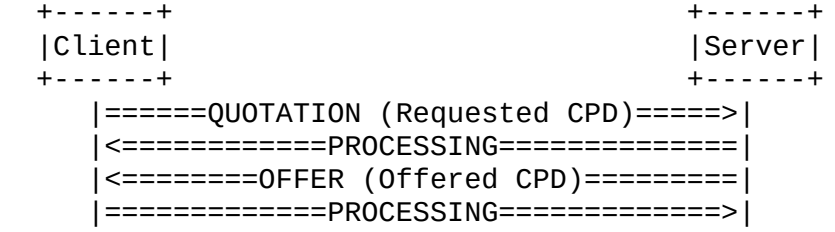
CPNP makes use of several flavors of Connectivity Provisioning Documents (CPD). These documents follow the CPP template described in [RFC7297].

Requested Connectivity Provisioning Document (Requested CPD): Refers to the CPD included by a CPNP client in a QUOTATION request.

Offered Connectivity Provisioning Document (Offered CPD): This document is included by a CPNP server in an OFFER message. Its information reflects the proposal of the server to accommodate all or a subset of the clauses depicted in a Requested CPD. A validity time is associated with the offer made.

Agreed Connectivity Provisioning Document (Agreed CPD): If the client accepts an offer made by the server, the Offered CPD is included in an ACCEPT message. This CPD is also included in an ACK message. Thus, a 3-way hand-shaking procedure is followed for successfully concluding the negotiation.

Figure 5 shows a typical CPNP negotiation cycle and the use of the different types of Connectivity Provisioning Documents.



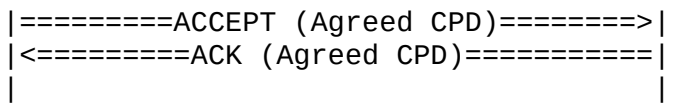


Figure 5: Connectivity Provisioning Documents

A provisioning document can include parameters with fixed values, loosely defined values, or a combination thereof. A provisioning document is said to be concrete if all clauses have fixed values.

A typical evolution of a negotiation cycle would start with a quotation order with loosely defined parameters, and then, as offers are made, it would conclude with concrete provisioning document for calling for the agreement.

8.8. Child Provisioning Quotation Orders

If the server detects that network resources from another Network Provider need to be allocated in order to accommodate the requirements described in a PQO (e.g., in the context of an inter-domain VPN service, additional PE router resources need to be allocated), the server may generate child PQOs to request the appropriate network provisioning operations (see Figure 6). In such situation, the server behaves also as a CPNP client. The server associates the parent order with its child PQOs. This is typically achieved by locally adding the reference of the child PQO to the parent order.

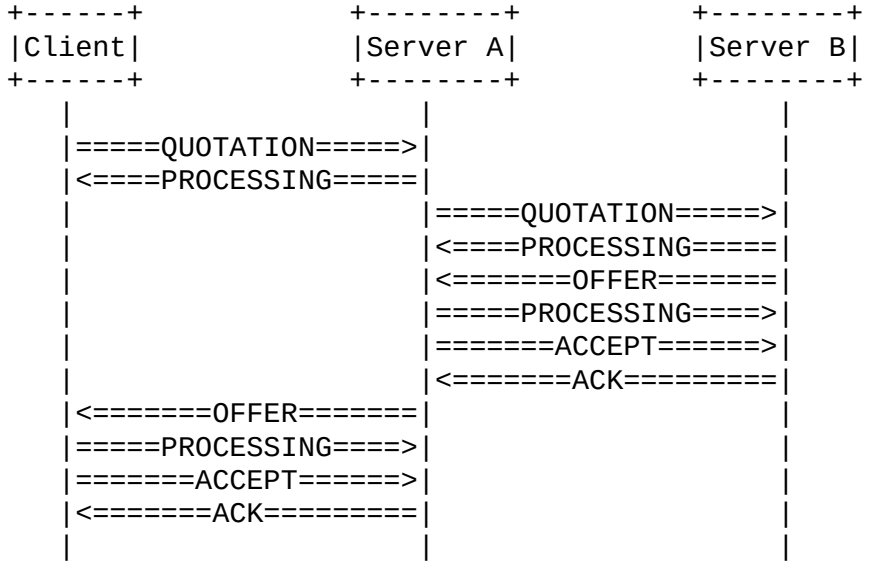


Figure 6: Example of Child Orders

8.9. Negotiating with Multiple CPNP Servers

A CPNP client may undertake multiple negotiations in parallel with several servers for practical reasons such as cost optimization and fail-safety. The multiple negotiations may lead to one or many agreements. Multiple negotiations with the same Provider are not precluded.

The salient point underlining the parallel negotiations scenario is that although the negotiation protocol is strictly between two parties, the negotiation logic may not necessarily be. The CPNP client negotiation logic may need to collectively drive parallel negotiations, as the negotiation with one server may affect the

negotiation with other servers; e.g., it may need to use the responses from all servers as input for determining the messages (and their content) to subsequently send in each individual negotiation. Timing is therefore an important aspect at the client's. The CPNP client needs to have the ability to synchronize the receipt of the responses from the servers. CPNP takes into account this requirement by allowing clients to specify in the QUOTATION message the time by which the server needs to respond (see Section 9.1.6).

8.10. State Management

Both the client and the server maintain repositories to store on-going orders. How these repositories are maintained is deployment-specific. It is out of scope of this document to elaborate on such considerations. Timestamps are also logged to track state change.

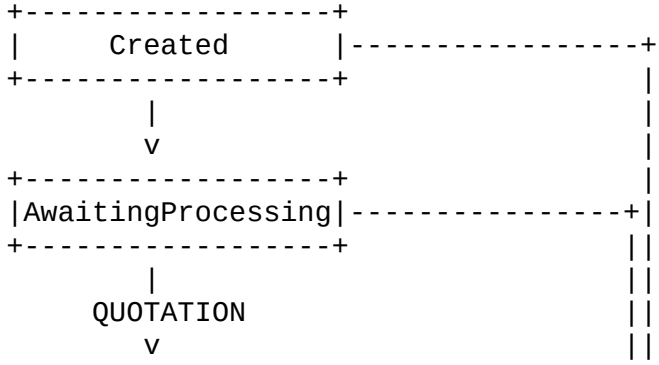
Tracking may be needed for various reasons, including regulatory ones.

In order to accommodate failures that may lead to the reboot of the client or the server, the use of permanent storage is recommended, thereby facilitating state recovery.

8.10.1. On the Client Side

The following lists the states which can be associated with a given order on the client's side:

- Created: when the order has been created. It is not handled by the client until the administrator allows to process it.
- AwaitingProcessing: when the administrator approved of processing a created order and the order has not been handled yet.
- PQOSent: when the order has been sent to the server.
- ServerProcessing: when the server has confirmed the receipt of the order.
- OfferReceived: when an offer has been received from the server.
- OfferProcessing: when a received offer is currently processed by the client.
- AcceptSent: when the client confirmed the offer to the server.
- AcceptAck: when the offer is acknowledged by the server.
- Cancelled: when the order has failed or cancelled.



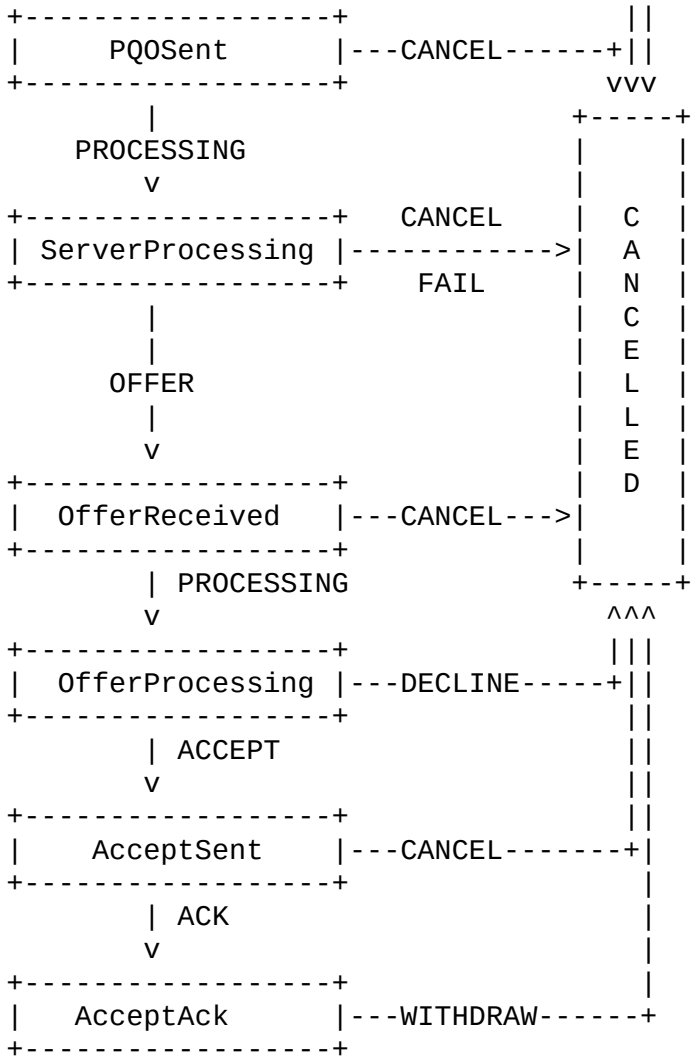


Figure 7: CPNP Finite State Machine (Client Side)

8.10.2. On the Server Side

The following lists the states which can be associated with a given order and a corresponding offer on the server's side:

- PQOReceived: when the order has been received from the client.
- AwaitingProcessing: when the order is being processed by the server. An action from the server administrator may be needed.
- OfferProposed: when the request has been successfully handled and an offer has been sent to the client.
- ProcessingReceived: when the server received a PROCESSING for an offer sent to the client.
- AcceptReceived: when the server received a confirmation for the offer from the client.
- AcceptAck: when the server acknowledged the offer (accepted by client) to the client.
- Cancelled: when the order has failed to be met or it has been cancelled by the client. Associate resources must be released in the latter case, if prior reserved.

- ChildCreated: when a child order has been created in cases where resources from another Network Provider are needed.

- ChildPQOSent: when a child order has been sent to the remote server.

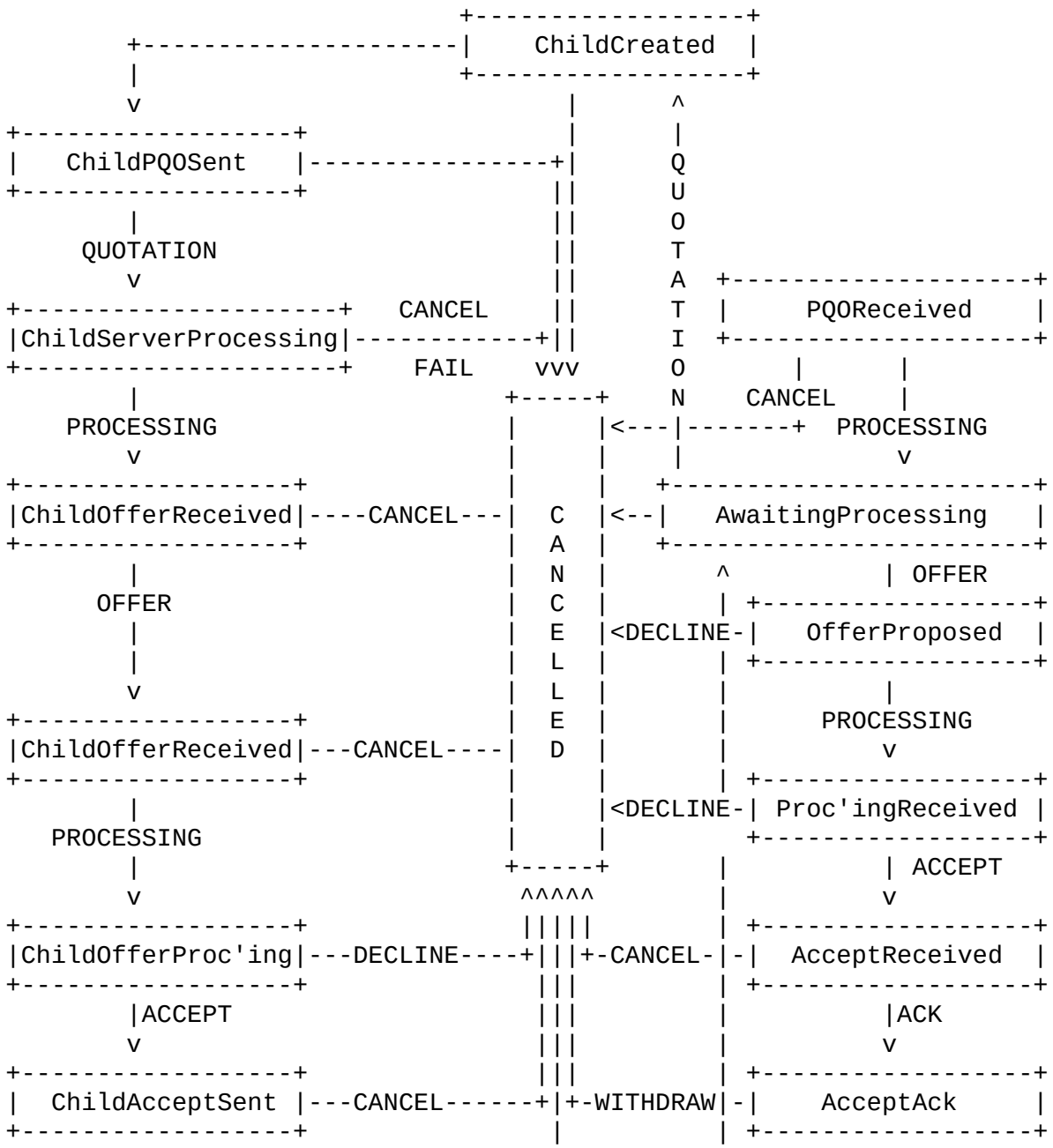
- ChildServerProcessing: when a child order is currently processed by the remote server.

- ChildOfferReceived: when an offer has been received to a child order from the remote server.

- ChildOfferProcessing: when a received offer to a child order is currently processed.

- ChildAcceptSent: when the child offer (offer received from the remote server in response to a child order) is confirmed to the remote server.

- ChildAcceptAck: when an accepted child offer is acknowledged by the remote server.



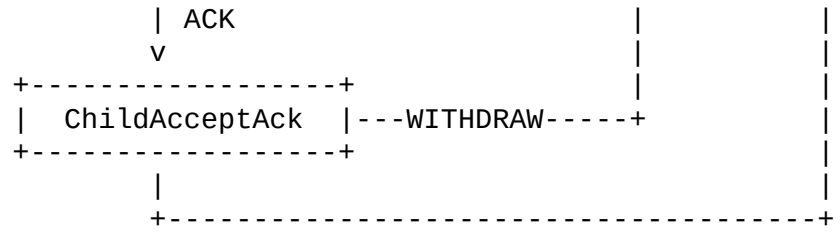


Figure 8: CPNP Finite State Machine (Server Side)

9. CPNP Objects

This section defines CPNP objects using the RBNF format defined at [RFC5511].

~~Note:~~

Note 1: The formats of CPNP messages are provided using a generic format. Implementors can adapt RBNF definitions to their "favorite" message format. For example, JSON [RFC8259] or **CBOR** [RFC7049] can be used.

Note 2: CPNP messages cannot be blindly mapped to RESTCONF messages with the target service being modelled as configuration data because such data is supposed to be manipulated by a RESTCONF client only. In such model, the RESTCONF server cannot use a value other than the one set by the client (e.g., Section 9.2.3) or remove offers from its own initiative (e.g., Section 9.1.8). An alternate approach might be to map CPNP operations into RESTCONF actions (rpc). Assessing the feasibility of such approach is out of scope.

9.1. Attributes

9.1.1. CUSTOMER_AGREEMENT_IDENTIFIER

CUSTOMER_AGREEMENT_IDENTIFIER is an identifier which is assigned by a client to identify an agreement. This identifier must be unique to the client.

Rules for assigning this identifier are specific to the client (Customer). The value of CUSTOMER_AGREEMENT_IDENTIFIER is included in all CPNP messages.

The client (Customer) assigns an identifier to an order under negotiation before an agreement is reached. This identifier will be used to unambiguously identify the resulting agreement at the client side (Customer).

The server handles CUSTOMER_AGREEMENT_IDENTIFIER as an opaque value.

9.1.2. PROVIDER_AGREEMENT_IDENTIFIER

PROVIDER_AGREEMENT_IDENTIFIER is an identifier which is assigned by a server to identify an order. This identifier must be unique to the server.

Rules for assigning this identifier are specific to the server (Provider). The value of PROVIDER_AGREEMENT_IDENTIFIER is included in all CPNP messages, except QUOTATION messages.

The server (Provider) assigns an identifier to an order under negotiation before an agreement is reached. This identifier will be used to unambiguously identify the resulting agreement at the server side (Provider).

The client handles PROVIDER_AGREEMENT_IDENTIFIER as an opaque value.

9.1.3. TRANSACTION_ID

This object conveys the Transaction-ID introduced in Section 8.4.

9.1.4. SEQUENCE_NUMBER

Sequence Number is a number that is monotonically incremented on every new CPNP message within a CPNP transaction. This number is used to avoid reply attacks.

Refer to Section 11.3.

9.1.5. NONCE

NONCE is a random value assigned by the CPNP server. It is RECOMMENDED to assign unique NONCE values for each order.

NONCE is then mandatory to be included in subsequent CPNP client operations on the associated order (including the resulting agreement) such as: withdraw the order or update the order.

If the NONCE validation checks fail, the server rejects the request with a FAIL message including the appropriate failure reason code.

9.1.6. EXPECTED_RESPONSE_TIME

This attribute indicates the time by when the CPNP client is expecting to receive a response, for a PQO, from the CPNP server. If no offer is received by then, the CPNP client will consider the quotation order as rejected.

EXPECTED_RESPONSE_TIME follows the date format specified in [RFC1123].

9.1.7. EXPECTED_OFFER_TIME

This attribute indicates the time by when the CPNP server is expecting to make an offer to the CPNP client. If no offer is received by then, the CPNP client will consider the order as rejected.

The CPNP server may propose an expected offer time that does not match the expected response time indicated in the quotation order message. The CPNP client can accept or rejects the proposed expected time by when the CPNP server will make an offer.

The CPNP server can always request extra time for its processing, but this may be accepted or rejected by the CPNP client.

EXPECTED_OFFER_TIME follows the date format specified in [RFC1123].

9.1.8. VALIDITY_OFFER_TIME

This attribute indicates the time of validity of an offer made by the

CPNP server. If the offer is not accepted before this date expires, the CPNP server will consider the CPNP client has rejected the offer; the CPNP server will silently clear this order.

VALIDITY_OFFER_TIME follows date format specified in [RFC1123].

9.1.9. CONNECTIVITY_PROVISIONING_DOCUMENT

The RBNF format of the Connectivity Provisioning Document (CPD) is shown in Figure 9:

```
<CONNECTIVITY_PROVISIONING_DOCUMENT> ::=
                            <Connectivity Provisioning Component> ...
<Connectivity Provisioning Component> ::=
                            <CONNECTIVITY_PROVISIONING_PROFILE> ...
<CONNECTIVITY_PROVISIONING_PROFILE> ::=
                            <Customer Nodes Map>
                            <SCOPE>
                            <QoS Guarantees>
                            <Availability>
                            <CAPACITY>
                            <Traffic Isolation>
                            <Conformance Traffic>
                            <Flow Identification>
                            <Overall Traffic Guarantees>
                            <Routing and Forwarding>
                            <Activation Means>
                            <Invocation Means>
                            <Notifications>
<Customer Nodes Map> ::=  <Customer Node> ...
<Customer Node> ::=  <IDENTIFIER>
                     <LINK_IDENTIFIER>
                     <LOCALISATION>
```

Figure 9: The RBNF format of the Connectivity Provisioning Document (CPD)

9.1.10. CPNP Information Elements

An Information Element (IE) is an optional object which can be included in a CPNP message.

9.1.10.1. Customer Description

The client may include administrative information such as:

- Name
- Contact Information

The format of this Information Element is as follows:

```
<Customer Description> ::= [<NAME>] [<Contact Information>]
<Contact Information> ::=  [<EMAIL_ADDRESS>] [<POSTAL_ADDRESS>]
                           [<TELEPHONE_NUMBER> ...]
```

9.1.10.2. Provider Description

The server may include administrative information in an offer such as:

- Name

- o AS Number ([RFC6793])
- o Contact Information

The format of this Information Element is as follows:

```
<Provider Description> ::= [<NAME>] [<Contact Information>] [<AS_NUMBER>]
```

9.1.10.3. Negotiation Options

The client may include some negotiation options such as:

- o Setup purpose: A client may request to setup a connectivity only for testing purposes during a limited period. The order can be extended to become permanent if the client was satisfied during the test period. This operation is achieved using the UPDATE method.

The format of this Information Element is as follows:

```
<Negotiation Options> ::= [<PURPOSE>]
```

9.2. Operation Messages

This section specifies the RBNF format of CPNP operation messages. The following operation codes are used:

1: QUOTATION (Section 9.2.1)
2: PROCESSING (Section 9.2.2)
3: OFFER (Section 9.2.3)
4: ACCEPT (Section 9.2.4)
5: DECLINE (Section 9.2.5)
6: ACK (Section 9.2.6)
7: CANCEL (Section 9.2.7)
8: WITHDRAW (Section 9.2.8)
9: UPDATE (Section 9.2.9)
10: FAIL (Section 9.2.10)

9.2.1. QUOTATION

The format of the QUOTATION message is shown below:

```
<QUOTATION Message> ::=  <VERSION>
                         <METHOD_CODE>
                         <SEQUENCE_NUMBER>
                         <TRANSACTION_ID>
                         <CUSTOMER_AGREEMENT_IDENTIFIER>
                         [<EXPECTED_RESPONSE_TIME>]
                         <REQUESTED_CONNECTIVITY_PROVISIONING_DOCUMENT>
                         [<INFORMATION_ELEMENT>...]
```

A QUOTATION message MUST include an order identifier which is generated by the client. Because several orders can be issued to several servers, the QUOTATION message MUST also include a Transaction-ID.

The message MAY include an EXPECTED_RESPONSE_TIME which indicates by when the client is expecting to receive an offer from the server. QUOTATION message MUST also include a requested connectivity provisioning document.

When the client sends the QUOTATION message to the server, the state

of the order changes to "PQOSent".

9.2.2. PROCESSING

The format of the PROCESSING message is shown below:

```
<PROCESSING Message> ::= <VERSION>
                         <METHOD_CODE>
                         <SEQUENCE_NUMBER>
                         <TRANSACTION_ID>
                         <CUSTOMER_AGREEMENT_IDENTIFIER>
                         <PROVIDER_AGREEMENT_IDENTIFIER>
                         [<EXPECTED_OFFER_TIME>]
```

Upon receipt of a QUOTATION message, the server proceeds with parsing rules (see Section 10). If no error is encountered, the server generates a PROCESSING response to the client to indicate the PQO has been received and it is being processed. The server MUST generate an order identifier which identifies the order in its local order repository. The server MUST copy the content of CUSTOMER_AGREEMENT_IDENTIFIER and TRANSACTION_ID fields as conveyed in the QUOTATION message. The server MAY include an EXPECTED_OFFER_TIME by when it expects to make an offer to the client.

Upon receipt of a PROCESSING message, the client verifies whether it has issued a PQO to that server and which contains the CUSTOMER_AGREEMENT_IDENTIFIER and TRANSACTION_ID. If no such PQO is found, the PROCESSING message MUST be silently ignored. If a PQO is found, the client may check if it accepts the EXPECTED_OFFER_TIME and then, it changes to state of the order to "ServerProcessing".

If more time is required by the server to process the quotation order, it MAY send a PROCESSING message that includes a new EXPECTED_OFFER_TIME. The client can answer with an ACK message if more time is granted (Figure 10) or with a FAIL message if the time extension is rejected (Figure 11).

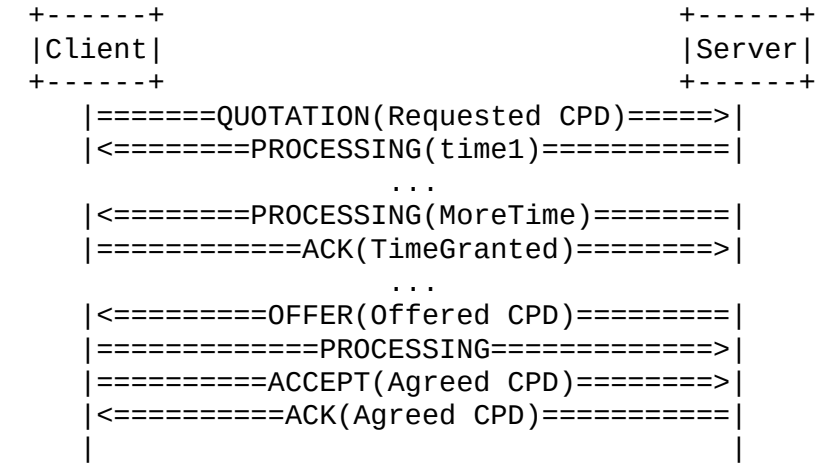
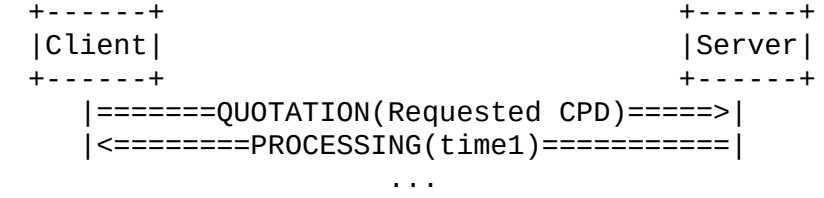


Figure 10: Request More Negotiation Time: Granted



```
                |<========PROCESSING(MoreTime)========|
                |===========FAIL(TimeRejected)=======>|
```

Figure 11: Request More Negotiation Time: Rejected

9.2.3. OFFER

The format of the OFFER message is shown below:

```
<OFFER Message> ::= <VERSION>
                    <METHOD_CODE>
                    <SEQUENCE_NUMBER>
                    <TRANSACTION_ID>
                    <CUSTOMER_AGREEMENT_IDENTIFIER>
                    <PROVIDER_AGREEMENT_IDENTIFIER>
                    <NONCE>
                    <VALIDITY_OFFER_TIME>
                    <OFFERED_CONNECTIVITY_PROVISIONING_DOCUMENT>
                    [<INFORMATION_ELEMENT>...]
```

The server answers with an OFFER message to a QUOTATION request received from the client. The offer will be considered as rejected by the client if no confirmation (ACCEPT message sent by the client) is received by the server before the expiration of the validity time.

9.2.4. ACCEPT

The format of the ACCEPT message is shown below:

```
<ACCEPT Message> ::=  <VERSION>
                      <METHOD_CODE>
                      <SEQUENCE_NUMBER>
                      <TRANSACTION_ID>
                      <CUSTOMER_AGREEMENT_IDENTIFIER>
                      <PROVIDER_AGREEMENT_IDENTIFIER>
                      <NONCE>
                      <AGREED_CONNECTIVITY_PROVISIONING_DOCUMENT>
                      [<INFORMATION_ELEMENT>...]
```

This message is used by a client to confirm the acceptance of an offer received from a server. The fields of this message MUST be copied from the received OFFER message.

9.2.5. DECLINE

The format of the DECLINE message is shown below:

```
<DECLINE Message> ::=  <VERSION>
                       <METHOD_CODE>
                       <SEQUENCE_NUMBER>
                       <TRANSACTION_ID>
                       <CUSTOMER_AGREEMENT_IDENTIFIER>
                       <PROVIDER_AGREEMENT_IDENTIFIER>
                       <NONCE>
```

The client may issue a DECLINE message to reject an offer. CUSTOMER_AGREEMENT_IDENTIFIER, PROVIDER_AGREEMENT_IDENTIFIER, TRANSACTION_ID, and NONCE are used by the server as keys to find the corresponding order. If an order matches, the server changes the state of this order to "Cancelled" and then returns an ACK with a copy of the requested CPD to the requesting client.

If no order is found, the server returns a FAIL message to the requesting client.

A flow example is shown in Figure 12.

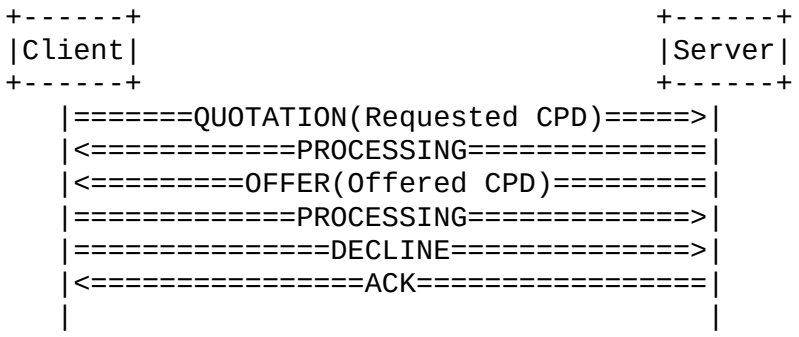


Figure 12: DECLINE Flow Example

9.2.6. ACK

The format of the ACK message is shown below:

```
<ACK Message> ::=  <VERSION>
                   <METHOD_CODE>
                   <SEQUENCE_NUMBER>
                   <TRANSACTION_ID>
                   <CUSTOMER_AGREEMENT_IDENTIFIER>
                   <PROVIDER_AGREEMENT_IDENTIFIER>
                   [<EXPECTED_RESPONSE_TIME>]
                   [<CONNECTIVITY_PROVISIONING_DOCUMENT>]
                   [<INFORMATION_ELEMENT>...]
```

This message is issued by the server to close a CPNP transaction or by a client to grant more negotiation time to the server.

This message is sent by the server as a response to an ACCEPT, WITHDRAW, DECLINE, or CANCEL message. In such case, the ACK message MUST include the copy of the Connectivity Provisioning Document as stored by the server, in particular:

- A copy of the requested/offered CPD is included by the server if it successfully handled a CANCEL message.
- A copy of the updated CPD is included by the server if it successfully handled an UPDATE message.
- A copy of the offered CPD is included by the server if it successfully handled an ACCEPT message in the context of a QUOTATION transaction.
- An empty CPD is included by the server if it successfully handled a DECLINE message.

A client may issue an ACK message as a response to a more time request (conveyed in PROCESSING) received from the server. In such case, the ACK message MUST include an EXPECTED_RESPONSE_TIME that is likely to be set to the time extension requested by the server.

9.2.7. CANCEL

The format of the CANCEL message is shown below:

```
<CANCEL Message> ::= <VERSION>
```

```
                        <METHOD_CODE>
                        <SEQUENCE_NUMBER>
                        <TRANSACTION_ID>
                        <CUSTOMER_AGREEMENT_IDENTIFIER>
                        [<CONNECTIVITY_PROVISIONING_DOCUMENT>]
```

The client can issue a CANCEL message at any stage during the CPNP negotiation process before an agreement is reached. CUSTOMER_AGREEMENT_IDENTIFIER and TRANSACTION_ID are used by the server as keys to find the corresponding order. If a quotation order matches, the server changes the state of this quotation order to "Cancelled" and then returns an ACK with a copy of the requested CPD to the requesting client.

If no quotation order is found, the server returns a FAIL message to the requesting client.

9.2.8. WITHDRAW

The format of the WITHDRAW message is shown below:

```
<WITHDRAW Message> ::= <VERSION>
                       <METHOD_CODE>
                       <SEQUENCE_NUMBER>
                       <TRANSACTION_ID>
                       <CUSTOMER_AGREEMENT_IDENTIFIER>
                       <PROVIDER_AGREEMENT_IDENTIFIER>
                       <NONCE>
                       [<AGREED_CONNECTIVITY_PROVISIONING_DOCUMENT>]
                       [<INFORMATION_ELEMENT>...]
```

This message is used to withdraw an offer already subscribed by the Customer. Figure 13 shows a typical usage of this message.

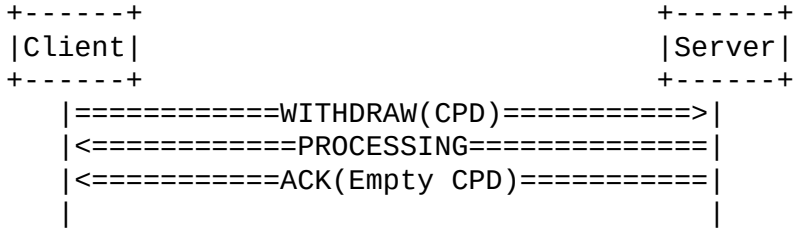


Figure 13: WITHDRAW Flow Example

The CPNP MUST include the same CUSTOMER_AGREEMENT_IDENTIFIER, PROVIDER_AGREEMENT_IDENTIFIER, and NONCE as those used when creating the order.

Upon receipt of a WITHDRAW message, the server checks whether an order matching the request is found. If an order is found, the state of the order is changed to "Cancelled" and an ACK message including an Empty CPD is returned to the requesting client. If no order is found, the server returns a FAIL message to the requesting client.

9.2.9. UPDATE

The format of the UPDATE message is shown below:

```
<UPDATE Message> ::= <VERSION>
                     <METHOD_CODE>
                     <SEQUENCE_NUMBER>
```

```
                        <TRANSACTION_ID>
                        <CUSTOMER_AGREEMENT_IDENTIFIER>
                        <PROVIDER_AGREEMENT_IDENTIFIER>
                        <NONCE>
                        <EXPECTED_RESPONSE_TIME>
                        <REQUESTED_CONNECTIVITY_PROVISIONING_DOCUMENT>
                        [<INFORMATION_ELEMENT>...]
```

This message is sent by the CPNP client to update an existing connectivity provisioning agreement. The CPNP MUST include the same CUSTOMER_AGREEMENT_IDENTIFIER, PROVIDER_AGREEMENT_IDENTIFIER, and NONCE as those used when creating the order. The CPNP client includes a new CPD which integrates the requested modifications. A new Transaction_ID MUST be assigned by the client.

Upon receipt of an UPDATE message, the server checks whether an order, having state "Completed", matches CUSTOMER_AGREEMENT_IDENTIFIER, PROVIDER_AGREEMENT_IDENTIFIER, and NONCE.

- If no order is found, the CPNP server generates a FAIL error with the appropriate error code.
- If an order is found, the server checks whether it can honor the request:
  - A FAIL message is sent to the client if the server cannot honor the request. The client may initiate a new PQO negotiation cycle.
  - An OFFER message including the updated connectivity provisioning document is sent to the client. For example, the server maintains an order for provisioning a VPN service that connects sites A, B and C. If the client sends an UPDATE message to remove site C, only sites A and B will be included in the OFFER sent by the server to the requesting client.

A flow chart that illustrates the use of UPDATE operation is shown in Figure 14.

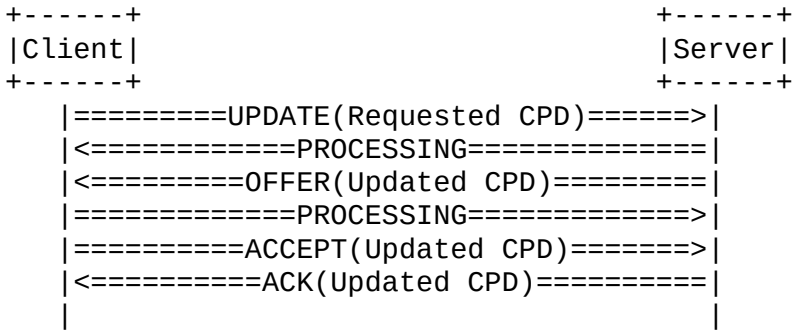


Figure 14: UPDATE Flow Example

9.2.10. FAIL

The format of the FAIL message is shown below:

```
<FAIL Message> ::= <VERSION>
                   <METHOD_CODE>
                   <SEQUENCE_NUMBER>
                   <TRANSACTION_ID>
                   <CUSTOMER_AGREEMENT_IDENTIFIER>
                   <PROVIDER_AGREEMENT_IDENTIFIER>
```

<STATUS_CODE>

This message is sent in the following cases:

- The server can not honor an order received from the client (i.e., received in a QUOTATION or UPDATE request).

- The server encounters an error when processing a CPNP request received from the client.
- The client can not grant more time to a the server. This is a response to a more time request conveyed in a PROCESSING message.

The status code indicates the error code. The following codes are supported:

1 (Message Validation Error):
The message can not be validated (see Section 10).
2 (Authentication Required):
The request cannot be handled because authentication is required.
3 (Authorization Failed):
The request cannot be handled because authorization failed.
4 (Administratively prohibited):
The request can not be handled because of administrative policies.
5 (Out of Resources):
The request can not be honored because there is not enough capacity.
6 (Network Presence Error):
The request can not be honored because there is no network presence.
7 (More Time Rejected):
The request to extend the time negotiation is rejected by the client.

10. CPNP Message Validation

Both client and server proceed with CPNP message validation. The following tables summarize the validation checks to be followed.

10.1. On the Client Side

Operation	Validation Checks
PROCESSING	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-ID,~~ **Transaction-ID,** Customer Order Identifier} must match an existing PQO with a state set to "PQOSent". The sequence number carried in the packet must be larger than the sequence number maintained by the client.
OFFER	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-ID,~~ **Transaction-ID,** Customer Order Identifier} must match an existing order with state set to "PQOSent" or {Source IP address, source port number, destination IP address, destination port number, Transaction-ID, Customer Order Identifier, Provider Order Identifier} must match an existing order with a state set to "ServerProcessing". The sequence number carried in the packet must be larger than the sequence number maintained by the client.

ACK (QUOTATION Transaction)	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-~~ **Transaction-ID,** ~~ID,~~ Customer Order Identifier, Provider Order Identifier, Offered Connectivity Provisioning Order} must match an order with a state set to "AcceptSent". The sequence number carried in the packet must be larger than the sequence number maintained by the client.
ACK (UPDATE Transaction)	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-ID,~~ **Transaction-ID,** Customer Order Identifier, Provider Order Identifier, Updated Connectivity Provisioning Order} must match an order with a state set to "AcceptSent". The sequence number carried in the packet must be larger than the sequence number maintained by the client.
ACK (WITHDRAW Transaction)	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-~~ **Transaction-ID,** ~~ID,~~ Customer Order Identifier, Provider Order Identifier, Empty Connectivity Provisioning Order} must match an order with a state set to "Cancelled". The sequence number carried in the packet must be larger than the sequence number maintained by the client.

10.2. On the Server Side

Method	Validation Checks
QUOTATION	The source IP address passes existing access filters (if any). The sequence number carried in the packet must not be less than the sequence number maintained by the server.
PROCESSING	The sequence number carried in the packet must be larger than the sequence number maintained by the server.
ACCEPT	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-ID,~~ **Transaction-ID,** Customer Order Identifier, Provider Order Identifier, Nonce, Offered Connectivity Provisioning Order} must match an order with state set to "OfferProposed" or "ProcessngReceived". The sequence number carried in the packet must be larger than the sequence number maintained by the server.
DECLINE	{Source IP address, source port number, destination IP address, destination port number, ~~Transaction-ID,~~ **Transaction-ID,** Customer Order Identifier, Provider Order Identifier, Nonce} must match an order with state set to "OfferProposed" or "ProcessngReceived". The sequence number carried in the packet must be larger than the sequence number maintained by the server.
UPDATE	The source IP address passes existing access filters (if any) and {Customer Order Identifier, Provider Order Identifier, Nonce} must match an existing order with state "Completed".
WITHDRAW	The source IP address passes existing access filters (if any) and {Customer Order Identifier, Provider Order Identifier, Nonce} must match an existing order with state "Completed".

11. Theory of Operation

Both CPNP client and server proceed to message validation checks as specified in Section 10.

11.1. Client Behavior

11.1.1. Order Negotiation Cycle

To place a provisioning quotation order, the client initiates first a local quotation order object identified by a unique identifier assigned by the client. The state of the quotation order is set to "Created". The client then generates a QUOTATION request which includes the assigned identifier, possibly an expected response time, a Transaction-ID and a Requested Connectivity Provisioning Document. The client may include additional Information Elements such as Negotiation Options.

The client may be configured to not enforce negotiation checks on EXPECTED_OFFER_TIME; if so no EXPECTED_RESPONSE_TIME attribute (or EXPECTED_RESPONSE_TIME set to infinite) should be included in the quotation order.

Once the request is sent to the server, the state of the request is set to "PQOSent" and a timer, if a response time is included in the quotation order, is set to the expiration time as included in the QUOTATION request. The client also maintains a copy of the CPNP session entry details used to generate the QUOTATION request. The CPNP client must listen on the same port number that it used to send the QUOTATION request.

If no answer is received from the server before the retransmission timer expires (i.e., RETRANS_TIMER, Section 8.5), the client proceeds to retransmission until maximum retry is reached (e.g., 3 times). The same sequence number is used for retransmitted packets.

If a FAIL message is received, the client may decide to issue another (corrected) request towards the same server, cancel the local order, or contact another server. The behavior of the client depends on the error code returned by the server in the FAIL message.

If a PROCESSING message matching the CPNP session entry (Section 8.3) is received, the client updates the CPNP session entry with the PROVIDER_AGREEMENT_IDENTIFIER information. If the client does not accept the expected offer time that may have been indicated in the PROCESSING message, the client may decide to cancel the quotation order. If the client accepts the EXPECTED_OFFER_TIME, it changes the state of the order to "ServerProcessing" and sets a timer to the value of EXPECTED_OFFER_TIME. If no offer is made before the timer expires, the client changes the state of the order to "Cancelled".

As a response to a more time request (conveyed in a PROCESSING message that included a new EXPECTED_OFFER_TIME), the client may grant this extension by issuing an ACK message or reject the time extension with a FAIL message having a status code set to "More Time Rejected".

If an OFFER message matching the CPNP session entry is received, the client checks if a PROCESSING message having the same PROVIDER_AGREEMENT_IDENTIFIER has been received from the server. If a PROCESSING message was already received for the same order but the PROVIDER_AGREEMENT_IDENTIFIER does not match the identifier included in the OFFER message, the client ignores silently the message. If a PROCESSING message having the same PROVIDER_AGREEMENT_IDENTIFIER was already received and matches the CPNP transaction identifier, the

client changes the state of the order to "OfferReceived" and sets a timer to the value of VALIDITY_OFFER_TIME indicated in the OFFER message.

If an offer is received from the server (i.e., as documented in an OFFER message), the client may accept or reject the offer. The client accepts the offer by generating an ACCEPT message which confirms that the client agrees to subscribe to the offer documented in the OFFER message; the state of the order is passed to "AcceptSent". The transaction is terminated if an ACK message is received from the server. If no ACK is received from the server, the client proceeds with the re-transmission of the ACCEPT message.

The client may also decide to reject the offer by sending a DECLINE message. The state of the order is set by the client to "Cancelled". If an offer is not acceptable by the client, the client may decide to contact a new server or submit another order to the same server. Guidelines to issue an updated order or terminate the negotiation are specific to the client.

11.1.2. Order Withdrawal Cycle

A client may withdraw a completed order. This is achieved by issuing a WITHDRAW message. This message MUST include Customer Order Identifier, Provider Identifier, and Nonce returned during the order negotiation cycle specified in Section 11.1.1.

If no ACK is received from the server, the client proceeds with the re-transmission of the message.

11.1.3. Order Update Cycle

A client may update a completed order. This is achieved by issuing an UPDATE message. This message MUST include Customer Order Identifier, Provider Order Identifier and Nonce returned during the order negotiation cycle specified in Section 11.1.1. The client MUST include in the UPDATE message an updated CPD with the requested changes.

Subsequent messages exchange is similar to what is documented in Section 11.1.1.

11.2. Server Behavior

11.2.1. Order Processing

Upon receipt of a QUOTATION message from a client, the server sets a CPNP session, stores Transaction-ID and generates a Provider Order Identifier. Once preliminary validation checks are completed (Section 10), the server may return a PROCESSING message to notify the client the quotation order is received and it is under processing; the server may include an expected offer time to notify the client by when an offer will be proposed. An order with state "AwaitingProcessing" is created by the server. The server runs its decision-making process to decide which offer it can make to honor the received order. The offer should be made before the expected offer time expires.

If the server cannot make an offer, it sends backs a FAIL message with the appropriate error code.

If the server requires more negotiation time, it must send a PROCESSING message with a new EXPECTED_OFFER_TIME. The client may grant this extension by issuing an ACK message or reject the time extension with a FAIL message having a status code set to "More Time Rejected". If the client doesn't grant more time, the server must answer before the initial expected offer time; otherwise the client will ignore the quotation order.

If the server can honor the request or it can make an offer that meet some of the requirements, it creates an OFFER message. The server must indicate the Transaction-ID, Customer Order Identifier as indicated in the QUOTATION message, and the Provider Order Identifier generated for this order. The server must also include Nonce and the offered Connectivity Provisioning Document. The server includes an offer validity time as well. Once sent to the client, the server changes the state of the order to "OfferSent" and a timer set to the validity time is initiated.

If the server determines that additional network resources from another network provider are needed to accommodate a quotation order, it will create child PQO(s) and will behave as a CPNP client to negotiate child PQO(s) with possible partnering providers (see Figure 6).

If no PROCESSING, ACCEPT or DECLINE message is received before the expiry of the RETRANS_TIMER, the server re-sends the same offer to the client. This procedure is repeated until maximum retry is reached.

If an ACCEPT message is received before the offered validity time expires, the server proceeds with validation checks as specified in Section 10. The state of the corresponding order is passed to "AcceptReceived". The server sends back an ACK message to terminate the order processing cycle.

If a CANCEL/DECLINE message is received, the server proceeds with the cancellation of the order. The state of the order is then passed to "Cancelled".

11.2.2. Order Withdrawal

A client may withdraw a completed order by issuing a WITHDRAW message. Upon receipt of a WITHDRAW message, the server proceeds with the validation checks, as specified in Section 10:

- If the checks fail, a FAIL message is sent back to the client with the appropriate error code.

- If the checks succeed, the server clears the clauses of the Connectivity Provisioning Document, changes the state of the order to "Cancelled", and sends back an ACK message with an Empty Connectivity Provisioning Document.

11.2.3. Order Update

A client may update an order by issuing an UPDATE message. Upon receipt of an UPDATE message, the server proceeds with the validation checks as specified in Section 10:

- If the checks fail, a FAIL message is sent back to the client with the appropriate error code.

- Subsequent messages exchange is similar to what is specified in Section 11.1.1. The server should generate a new Nonce value to be included in the offer made to the client.

11.3. Sequence Numbers

In each transaction, sequence numbers are used to protect the transaction against replay attacks. Each communicating partner of the transaction maintains two sequence numbers, one for incoming packets and one for outgoing packets. When a partner receives a message, it will check whether the sequence number in the message is larger than the incoming sequence number maintained locally. If not, the messages will be discarded. If the message is proved to be legal, the value of the incoming sequence number will be replaced by the value of the sequence number in the message. When a partner sends out a message, it will insert the value of outgoing sequence number into the message and increase the outgoing sequence number maintained locally by 1.

11.4. Message Re-Transmission

If a transaction partner sends out a message and does not receive any expected reply before the retransmission timer expires (i.e., RETRANS_TIMER), a transaction partner will try to re-transit the messages. An exception is the last message (e.g., ACK) sent from the server in a transaction. After sending this message, the retransmission timer will be disabled since no additional feedback is expected.

In addition, if the partner receives a re-sent last incoming packet, the partner can also send out the answer to the incoming packet with a limited frequency. If no answer was generated at the moment, the partner needs to generate a PROCESSING message as the answer.

To benefit message re-transmission, a partner could also store the last incoming packet and the associated answer. Note that the times of re-transmission could be decided by the local policy and re-transmission will not cause any change of sequence numbers.

12. Some Operational Guidelines

12.1. Logging on the CPNP Server

The CPNP server should be configurable to log various events and associated information. Such information may include:

- Client's IP address
- Any event change (e.g., new quotation order, offer sent, order confirm, order cancellation, order withdraw, etc.)
- Timestamp

12.2. Business Guidelines & Objectives

The CPNP server can operate in the following modes:

1. Fully automated mode:

   The CPNP server is provisioned with a set of business guidelines and objectives that will be used as an input to the decision-making process. The CPNP server will service received orders that falls into these business guidelines; otherwise requests

will be escalated to an administrator that will formally validate/invalidate an order request. The set of policies to be configured to the CPNP server are specific to each administrative entity managing a CPNP server.

2. Administrative-based mode:

This mode assumes some or all CPNP server' operations are subject to a formal administrative validation. CPNP events will trigger appropriate validation requests that will be forwarded to the contact person(s) or department which is responsible for validating the orders. Administrative validation messages are relayed using another protocol (e.g., SMTP) or a dedicated tool.

Business guidelines are local to each administrative entity. How validation requests are presented to an administrator are out of scope of this document; each administrative entity may decide the appropriate mechanism to enable for that purpose.

13. Security Considerations

Means to defend the server against denial-of-service attacks must be enabled. For example, access control lists (ACLs) can be enforced on the client, the server or the network in between, to allow a trusted client to communicate with a trusted server.

The client and the server MUST be mutually authenticated. Authenticated encryption MUST be used for data confidentiality and message integrity.

The protocol does not provide security mechanisms to protect the confidentiality and integrity of the packets transported between the client and the server. An underlying security protocol such as (e.g., Datagram Transport Layer Security (DTLS) [RFC6347], Transport Layer Security (TLS) [RFC8446]) MUST be used to protect the integrity and confidentiality for the protocol. In this case, if it is possible to provide an Automated Key Management (AKM) and associate each transaction with a different key, inter-transaction replay attacks can naturally be addressed. If the client and the server use a single key, an additional mechanism should be provided to protect inter-transaction replay attacks between them. Clients MUST implement DTLS record replay detection (Section 3.3 of [RFC6347]) or an equivalent mechanism to protect against replay attacks.

DTLS and TLS with a cipher suite offering confidentiality protection and the guidance given in [RFC7525] MUST be followed to avoid attacks on (D)TLS.

The client MUST silently discard CPNP responses received from unknown CPNP servers. The use of a randomly generated Transaction-ID makes it hard to forge a response from a server with a spoofed IP address belonging to a legitimate CPNP server. Furthermore, CPNP demands that messages from the server must include correct identifiers of the orders. Two order identifiers are used: one generated by the client and a second one generated by the server.

The Provider MUST enforce means to protect privacy-related information included the documents (see Section 8.7) exchanged using CPNP messages [RFC6462]. In particular, this information MUST NOT be revealed to external parties without the consent of Customers. Providers should enforce policies to make Customer fingerprinting

difficult to achieve. For more discussion about privacy, refer to [RFC6462][RFC6973].

The Nonce and the Transaction ID attributes provide sufficient randomness and can effectively tolerate attacks raised by off-line adversaries, who do not have the capability of eavesdropping and intercepting the packets transported between the client and the server. Only authorized clients must be able to modify agreed CPNP orders. The use of a randomly generated Nonce by the server makes it hard to modify an agreement on behalf of a malicious third-party.

14. IANA Considerations

This document does not request any IANA action.

15. Acknowledgements

Thanks to Diego R. Lopez and A. Farrel for the comments.

16. References

16.1. Normative References

[RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <https://www.rfc-editor.org/info/rfc1123>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, DOI 10.17487/RFC4086, June 2005, <https://www.rfc-editor.org/info/rfc4086>.

[RFC5511] Farrel, A., "Routing Backus-Naur Form (RBNF): A Syntax Used to Form Encoding Rules in Various Routing Protocol Specifications", RFC 5511, DOI 10.17487/RFC5511, April 2009, <https://www.rfc-editor.org/info/rfc5511>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC7297] Boucadair, M., Jacquenet, C., and N. Wang, "IP Connectivity Provisioning Profile (CPP)", RFC 7297, DOI 10.17487/RFC7297, July 2014, <https://www.rfc-editor.org/info/rfc7297>.

[RFC7525] Sheffer, Y., Holz, R., and P. Saint-Andre, "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", BCP 195, RFC 7525, DOI 10.17487/RFC7525, May 2015, <https://www.rfc-editor.org/info/rfc7525>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

16.2. Informative References

[AGAVE] Boucadair, M., Georgatsos, P., Wang, N., Griffin, D., Pavlou, G., Howarth, M., and A. Elizondo, "The AGAVE Approach for Network Virtualization: Differentiated Services Delivery", April 2009, <https://rd.springer.com/article/10.1007/s12243-009-0103-4>.

[ETICS] EU FP7 ETICS Project, "Economics and Technologies of Inter-Carrier Services", January 2014, <https://www.ict-etics.eu/fileadmin/documents/news/ETICS_white_paper_final.pdf>.

[I-D.boucadair-lisp-idr-ms-discovery] Boucadair, M. and C. Jacquenet, "LISP Mapping Service Discovery at Large", draft-boucadair-lisp-idr-ms-discovery-01 (work in progress), March 2016.

[I-D.geng-netslices-architecture] 67, 4., Dong, J., Bryant, S., kiran.makhijani@huawei.com, k., Galis, A., Foy, X., and S. Kuklinski, "Network Slicing Architecture", draft-geng-netslices-architecture-02 (work in progress), July 2017.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <https://www.rfc-editor.org/info/rfc2782>.

[RFC3084] Chan, K., Seligson, J., Durham, D., Gai, S., McCloghrie, K., Herzog, S., Reichmeyer, F., Yavatkar, R., and A. Smith, "COPS Usage for Policy Provisioning (COPS-PR)", RFC 3084, DOI 10.17487/RFC3084, March 2001, <https://www.rfc-editor.org/info/rfc3084>.

[RFC4026] Andersson, L. and T. Madsen, "Provider Provisioned Virtual Private Network (VPN) Terminology", RFC 4026, DOI 10.17487/RFC4026, March 2005, <https://www.rfc-editor.org/info/rfc4026>.

[RFC4176] El Mghazli, Y., Ed., Nadeau, T., Boucadair, M., Chan, K., and A. Gonguet, "Framework for Layer 3 Virtual Private Networks (L3VPN) Operations and Management", RFC 4176, DOI 10.17487/RFC4176, October 2005, <https://www.rfc-editor.org/info/rfc4176>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC6462] Cooper, A., "Report from the Internet Privacy Workshop", RFC 6462, DOI 10.17487/RFC6462, January 2012, <https://www.rfc-editor.org/info/rfc6462>.

[RFC6574] Tschofenig, H. and J. Arkko, "Report from the Smart Object Workshop", RFC 6574, DOI 10.17487/RFC6574, April 2012, <https://www.rfc-editor.org/info/rfc6574>.

[RFC6770] Bertrand, G., Ed., Stephan, E., Burbridge, T., Eardley, P., Ma, K., and G. Watson, "Use Cases for Content Delivery Network Interconnection", RFC 6770, DOI 10.17487/RFC6770, November 2012, <https://www.rfc-editor.org/info/rfc6770>.

[RFC6793] Vohra, Q. and E. Chen, "BGP Support for Four-Octet Autonomous System (AS) Number Space", RFC 6793, DOI 10.17487/RFC6793, December 2012, <https://www.rfc-editor.org/info/rfc6793>.

[RFC6830] Farinacci, D., Fuller, V., Meyer, D., and D. Lewis, "The Locator/ID Separation Protocol (LISP)", RFC 6830, DOI 10.17487/RFC6830, January 2013, <https://www.rfc-editor.org/info/rfc6830>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7149] Boucadair, M. and C. Jacquenet, "Software-Defined Networking: A Perspective from within a Service Provider Environment", RFC 7149, DOI 10.17487/RFC7149, March 2014, <https://www.rfc-editor.org/info/rfc7149>.

[RFC7215] Jakab, L., Cabellos-Aparicio, A., Coras, F., Domingo-Pascual, J., and D. Lewis, "Locator/Identifier Separation Protocol (LISP) Network Element Deployment Considerations", RFC 7215, DOI 10.17487/RFC7215, April 2014, <https://www.rfc-editor.org/info/rfc7215>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7491] King, D. and A. Farrel, "A PCE-Based Architecture for Application-Based Network Operations", RFC 7491, DOI 10.17487/RFC7491, March 2015, <https://www.rfc-editor.org/info/rfc7491>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8299] Wu, Q., Ed., Litkowski, S., Tomotaki, L., and K. Ogaki, "YANG Data Model for L3VPN Service Delivery", RFC 8299,

DOI 10.17487/RFC8299, January 2018,
<https://www.rfc-editor.org/info/rfc8299>.

[RFC8309] Wu, Q., Liu, W., and A. Farrel, "Service Models Explained", RFC 8309, DOI 10.17487/RFC8309, January 2018, <https://www.rfc-editor.org/info/rfc8309>.

[RFC8329] Lopez, D., Lopez, E., Dunbar, L., Strassner, J., and R. Kumar, "Framework for Interface to Network Security Functions", RFC 8329, DOI 10.17487/RFC8329, February 2018, <https://www.rfc-editor.org/info/rfc8329>.

[RFC8597] Contreras, LM., Bernardos, CJ., Lopez, D., Boucadair, M., and P. Iovanna, "Cooperating Layered Architecture for Software-Defined Networking (CLAS)", RFC 8597, DOI 10.17487/RFC8597, May 2019, <https://www.rfc-editor.org/info/rfc8597>.

Authors' Addresses

Mohamed Boucadair
Orange
Rennes 35000
France

Email: mohamed.boucadair@orange.com

Christian Jacquenet
Orange
Rennes 35000
France

Email: christian.jacquenet@orange.com

Dacheng Zhang
Huawei Technologies

Email: dacheng.zhang@huawei.com
Panos Georgatsos
Centre for Research and Innovation Hellas
78, Filikis Etairias str.
Volos, Hellas 38334
Greece

Phone: +302421306070
Email: pgeorgat@gmail.com