Network Working Group
Internet-Draft
Intended status: Experimental
Expires: April 22, 2013

S. Randriamasy, Ed.
W. Roome
Alcatel-Lucent Bell Labs
N. Schwan

October 19, 2012

Multi-Cost ALTO
draft-randriamasy-alto-multi-cost-07

Abstract

IETF is designing a new service called ALTO (Application Layer traffic Optimization) that includes a "Network Map Service", an "Endpoint Cost Service" and an "Endpoint (EP) Ranking Service" and thus incentives for application clients to connect to ISP preferred Endpoints. These services provide a view of the Network Provider (NP) topology to overlay clients.

The present draft proposes a light way to extend the information provided by the current ALTO protocol in two ways. First, including information on multiple Cost Types in a single ALTO transaction provides a better mapping of the Selected Endpoints to needs of the growing diversity of Content and Resources Networking Applications and to the network conditions. Second, one ALTO query and response exchange on N Cost Types is faster and lighter than N single cost transactions. All this also helps producing a faster and more robust choice when multiple Endpoints need to be selected.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

IETF is designing a new service called ALTO that provides guidance to overlay applications, which have to select one or several hosts from a set of candidates that are able to provide a desired resource. This guidance is based on parameters that affect performance and efficiency of the data transmission between the hosts, e.g., the topological distance. The purpose of ALTO is to improve Quality of Experience (QoE) in the application while reducing resource consumption in the underlying network infrastructure. The ALTO protocol conveys the Internet View from the perspective of a Provider Network region that spans from a region to one or more Autonomous System (AS). Together with this Network Map, it provides the Provider determined Cost Map between locations of the Network Map. Last, it provides the Ranking of Endpoints w.r.t. their routing cost.

Current ALTO Costs and their modes provide values that are seen to be stable over a longer period of time, such as hopcount and administrative routing cost to reflect ISP routing preferences. Recently, new use cases have extended the usage scope of ALTO to Content Delivery Networks, Data centers and applications that need additional information to select their Endpoints or handle their PIDs.

Thus a multitude of new Cost Types that better reflect the requirements of these applications are expected to be specified, in particular cost values that change more frequently than previously assumed.

The current ALTO protocol draft [ID-alto-protocol-11] restricts ALTO Cost Maps and Endpoint Cost services to only one Cost Type and Cost Mode per ALTO request. To retrieve information for several Cost Types, an ALTO client must send several separate requests to the server.

It would be far more efficient, in terms of RTT, traffic, and processing load on the ALTO client and server, to get all costs with a single query/response transaction. Vector costs provide a robust and natural input to multi-variate path computation as well as robust multi-variate selection of multiple Endpoints. In particular, one Cost Map reporting on N Cost Types is less bulky than N Cost Maps containing one Cost Type each. This is valuable for both the storage of these maps and their transmission. Additionally, for many emerging applications that need information on several Cost Types, having them gathered in one map will save time.

There are three parts in this draft. The first part exposes use cases motivating the introduction of new Cost Types and why multi-

cost transactions are useful. The second part specifies the core ALTO protocol extensions that are required or recommended to support requests and responses on multiple Cost Types in one single transaction. These extensions also integrate the discussions within the ALTO Working Group. The third part lists the Multi-Cost ALTO services that can be supported by these extensions.

2. Application scope and terminology

This draft generalizes the case of a P2P client to include the case of a CDN client, a client of an application running on a virtual server, a GRID application client and any Client having the choice in several connection points for data or resource exchange. To do so, it uses the term "Application Client" (AC).

This draft focuses on the use case where the ALTO client is embedded in the Application Client or in some Application Endpoint tracker in the network, such as a P2P tracker, a CDN request router or a cloud computing orchestration system implemented in a logically centralized management system.

It is assumed that Applications likely to use the ALTO service have a choice in connection endpoints as it is the case for most of them. The ALTO service is managed by the Network Provider (NP) and reflects its preferences for the choice of endpoints. The NP defines in particular the network map, the routing cost among Network Locations, the cost types used to reflect it, and which ALTO services are available at a given ALTO server.

The draft uses terms defined as follows:

- Endpoint (EP): can be a Peer, a CDN storage location, a physical server involved in a virtual server-supported application, a Party in a resource sharing swarm such as a computation Grid or an online multi-party game.

- Endpoint Discovery (EP Discovery) : this term covers the different types of processes used to discover the eligible endpoints.

- Network Service Provider (NSP): includes both ISPs, who provide means to transport the data, and Content Delivery Networks (CDNs) who care for the dissemination, persistent storage and possibly identification of the best/closest content copy.

- ALTO transaction: a request/response exchange between an ALTO Client and an ALTO Server.

- Application Client (AC): this term generalizes the case of a P2P client to include the case of a CDN client, a client of an application running on a virtual server, a GRID application client and any Client having the choice in several connection points for data or resource exchange.

3. Uses cases for using multiple costs

The ALTO protocol specification in [ID-alto-protocol-11] focuses on the basic use case of optimizing routing costs in NSP networks. Upcoming use cases however will require both new Cost Types and new Endpoint Properties. Recent ALTO use cases now extend to CDNs, Data centers and other applications that need additional information to select their Endpoints or handle their PIDs. The needed Cost Types depend on the QoE requirements that are specific to the applications. Moreover, the cost values that they may use may change more rapidly than assumed up to now.

The goal of this section is to describe forward looking use case scenarios that are likely to benefit from ALTO, in order to motivate the introduction of new Cost Types and Endpoint Properties as well as the ALTO Multi-Cost extension.

3.1. Use cases for using additional costs

ALTO Cost Types and Endpoint Properties are registered in two registries maintained by IANA. The ALTO Cost Type registry ensures that the Cost Types that are represented by an ALTO Cost Map are unique identifiers, and it further contains references to the semantics of the Cost Type. The current specification registers 'routingcost' as a generic measure for routing traffic from a source to a destination. In a similar way the ALTO Endpoint Property Registry ensures uniqueness of ALTO Endpoint Property identifiers and provides references to particular semantics of the allocated Endpoint Properties. Currently the 'pid' identifier is registered, which serves as an identifier that allows aggregation of network endpoints into network regions. Both registries accept new entries after Expert Review. New entries should conform to the respective syntactical requirements, and must include information about the new identifier, the intended semantics, and the security considerations. One basic example advocating for multiple Cost Type transactions is an Application Client looking for destination Endpoints or Source/Destination PID pairs yielding jointly the lowest 'routingcost' and path delay. We hereby assume that 'routingcost' values report some monetary cost and that the Application Client chooses to rely on the hopcount to reflect the path delay.

3.1.1. Delay Sensitive Overlay Applications

The ALTO working group has been created to allow P2P applications and NSPs a mutual cooperation, in particular because P2P bulk file-transfer applications have created a huge amount of intra-domain and congestion on low-speed uplink traffic. By aligning overlay topologies according to the 'routingcost' of the underlying network, both layers are expected to benefit in terms of reduced costs and improved Quality-of-Experience.

Other types of overlay applications might benefit from a different set of path metrics. In particular for real-time sensitive applications, such as gaming, interactive video conferencing or medical services, creating an overlay topology with respect to a minimized delay is preferable. However it is very hard for an NSP to give accurate guidance for this kind of realtime information, instead probing through end-to-end measurements on the application layer has proven to be the superior mechanism. Still, a NSP might give some guidance to the overlay application, for example by providing statistically preferable paths, possibly with respect to the time of day. Also static information like hopcount can serve as an indicator for the delay that can be expected. Thus a Cost Type that can indicate latency, without the need for end-to-end measurements between endpoints, is likely to be useful.

3.1.2. Selection of physical servers involved in virtualized applications

Virtualized applications in large Datacenters are supported by virtualized servers that actually gather resources distributed on several physical servers. The federation of these resources is often orchestrated by a centralized entity that needs to select the physical servers from or to which it will take resources. This entity can be co-located with an ALTO Client that will request and get the ALTO information on the network formed by the physical servers. The physical servers can be assimilated to endpoints with which the orchestration entity trades application resources or content. These resources include computation resources, storage capacity and path bandwidth between the physical servers.

Here too, the applications that are ran are diverse and may have different and specific QoE requirements. The Endpoint selection typically needs to consider both the computational resources at the Endpoints and the resources e.g. in bandwidth on the transmission paths to or among Endpoints. Thus the application QoE requirements drive the Endpoint selection with more or less weight on QoE specific metrics such as hopcount/delay, bandwidth and other resources, that are typically combined with the routing cost and need to jointly

integrate the Endpoint and transmission path perspective in the decision process, which is difficult to do with one single Cost Type.

3.1.3. CDN Surrogate Selection

Another use case is motivated through draft [draft-jenkins-alto-cdn-use-cases-01]. The request router in today's CDNs makes a decision about the surrogate or cache node to which a content request should be forwarded. Typically this decision is based on locality aspects, i.e. the request router tries to select the surrogate node losest to the client. By using the 'routingcost' Cost Type, an ALTO server allows an NSP to guide the CDN in selecting the best cache node. This is particularly important as CDNs place cache nodes deeper into the network (i.e., closer to the end user), which requires finer grained information. Finally the provisioning of abstracted network topology information across administrative boundaries gains importance for cache federations.

While distance today is the predominant metric used for routing decisions, other metrics might allow sophisticated request routing strategies. For example the load a cache node sees in terms of CPU utilization, memory usage or bandwidth utilization might influence routing decisions for load-balancing reasons. There exist numerous ways of gathering and feeding this kind of information into the request routing mechanism.

For example, information reporting on the occupation level of a cache could be based on a cost reflecting: its remaining computation resources, its remaining storage capacity w.r.t its capacity in storage or computation resources.

As ALTO is likely to become a standardized interface to provide network topology information, the ALTO server could also provide other information that a request router needs. In the next iterations of this draft we will analyse which of these metrics is suitable as a Cost Type or Endpoint Property for CDN Surrogate Selection, and propose to register them in the respective registries.

3.1.4. Some proposed additional properties and costs

In addition to CDN caches, Endpoint Properties and Costs can be useful to report an Endpoint's load, given that an Endpoint can as well be a physical server in a datacenter or any entity as defined in Section 2 of this draft.

Proposed new Endpoint properties and costs include:

- an Endpoint Property called "EPCapacity", reflecting the nominal capacity of this endpoint. This capacity could be split into:

  * EP Nominal Memory: the storage capacity of the Endpoint.

  * EP Nominal Bandwidth: the capacity of the computation resources of the Endpoint.

- an Endpoint Cost called "EP occupied Capacity", reflecting the currently available resources w.r.t. their nominal capacity. As with EP Capacity, this can be split into:

  * EP Occupied Memory: the remaining storage capacity,

  * EP Occupied Bandwidth: the remaining computation resources.

Likewise, new Cost Types are needed to describe the resources of the network paths needed for content transport, in particular the utilized network path bandwidth.

- A Cost Type named 'pathoccupationcost' (POC) can be used to reflect the NP view of the utilized path bandwidth. Such an ALTO Cost Type is likely to have values that change frequently. By no means, as stated in the ALTO requirements, are ALTO Cost types expected to reflect real-time values, as these can be gathered by other mechanisms. Instead, a Cost Type such as 'pathoccupationcost' should be used as an abstraction that may be represented by a statistical value, or be updated regularly at a frequency lower than 'real-time', or be provided according to different time periods or other parameters. A provision mode for time dependent cost values is proposed in [draft-randriamasy-alto-cost-schedule-01]

3.2. Use cases for Multi-Cost ALTO transactions

Different Cost Types are suitable for different applications. For example, delay sensitive applications look for both low routing cost and low delay, where as other applications, such as non real time content download, look for moderate delay and minimal losses. On the other hand, applications or entities managing application input information may want, for various reasons to update their ALTO information on several Cost Types. So an ALTO Client may want to mix Cost Types in either 'numerical' and 'ordinal' mode, for Cost Types values that can be represented by numerical values.

The Multi-Cost ALTO Services propose to:

- include several Cost Types (and/or Cost Modes) in an ALTO client's Cost Map and Endpoint Cost request,

- provide several Cost Type values (and/or Cost Mode) in an ALTO server's response, instead of one.

The primary reasons to use Multi-Cost ALTO are:

- Optimizing time and bandwidth: a single ALTO response with a Multi-Cost cost map with three separate Cost Type values takes much less network bandwidth, and fewer CPU cycles, than three separate ALTO requests for three complete single-cost cost maps. The motivation also holds for the Endpoint Cost Service. Multi-Cost ALTO services can straightforwardly provide a more complete set of cost information.

- Facing unpredictable and/or rapid value changes: an ALTO client can get a consistent snapshot of several different rapidly-varying Cost Type values.

3.2.1. Optimized Endpoint Cost Service

The Endpoint Cost Service (ECS) provides cost information about both the application Endpoint resources and the networking resources used to access those Endpoints. In addition, the ECS may be invoked in "short term" situations, that is for frequent requests and/or requests requiring fast responses. For the ECS, the server's response is restricted to the requested Endpoints, and so is much smaller than a complete Cost Map. Therefore the ECS can be invoked for 'nearly-instant' information requests, and is particularly well suited for multi-cost ALTO transactions, supporting requests and responses on several Cost Type values simultaneously.

3.2.2. Optimized Filtered Cost Map Service

The set of ALTO Cost Types is not restricted to 'routingcost': ALTO Servers may provide a broader set of metrics. One thing to consider is that the frequency of updates can vary from a Cost Type to another one. Additionally the volume of an entire cost map with values of all available Cost Types, may get rapidly prohibitive for frequent downloads. Given these considerations the Application Client may take better advantage when:

- requesting multi-cost maps filtered w.r.t. Cost Types of compatible update frequencies or dates, which is the responsibility of the Application Client,

- requesting multi-cost maps filtered w.r.t. a restricted set of PID pairs.

In such a case, as with the Endpoint Cost Service, the purpose of a Multi-Cost transaction is to gain time with whatever future use of the received ALTO information. In this case, the Client may mix Cost Types in either 'numerical' and 'ordinal' mode, for Cost Type values that can be represented by numerical values.

3.2.3. Cases of unpredicable Endpoint cost value changes

Querying all Endpoint cost values simultaneously is always more time and resources efficient than doing it sequentially.

It becomes a necessity in case of unpredictable and/or rapid value changes on at least one of the ALTO Cost Types. The term 'rapid' here means "Typical update intervals [that] may be several orders of magnitude longer than the typical network-layer packet round-trip time (RTT)", as described in [ID-ALTO-Requirements13], up to a couple of minutes.

This section provides two examples of a delay sensitive application using 'routingcost' and 'hopcount' to select an Endpoint. The application can choose between two candidate Endpoints, EP1 and EP2. The initial choice at T=1 is EP1. It is assumed that at T=2 events in the network occur that impact both 'routingcost' and 'hopcount'.

These examples illustrate the need to query 'hopcount' and 'routingcost' values at the same time in order to re-evaluate the EP costs w.r.t. the QoE needs of the application. It is assumed that the application triggers regular ALTO requests to get the latest cost values for a list of candidate Endpoints.

In some cases the Application client wants to use the ALTO information to perform multi-variate optimization on several Cost Type values. In order for the optimization to be reliable, it is recommended that the Cost Type values are provided in 'numerical' Cost Mode. Therefore the requested Cost Mode for the applicable Cost Types SHOULD be 'numerical'.

3.2.3.1. Case of a Multi-Cost ALTO query upon a route change

In Figure 1, initially at time T=1, the application has chosen EP1 rather than EP2, despite the higher routing cost, because EP1 has a "better" (lower) 'hopcount' value and despite the higher routing cost and possibly because the application has set a higher weight to 'hopcount'.

At a time T=2, the route to EP1 changes. The ALTO Server information is accordingly updated. The ALTO client makes its next request to update the cost values for 'routingcost' and 'hopcount' on EP1 and EP2. It appears that EP1 has now a hopcount value of 3, the same than for EP2 while its routing cost is higher.

The application realizes that there is no more benefit in keeping interacting with EP1 and therefore switches to EP2, that now has the same hopcount but a lower routing cost.

```
T = 1 : EP1: routingcost = 40, hopcount = 2
         EP2: routingcost = 30, hopcount = 3

         EP1 is selected because application is time-sensitive and
         metric 'hopcount' has a higher weight

                                           .-----.
              O ---------- O ------------- | EP2 |
            /                              `-----'
          /
        /                               .-----.
  Source ----------------------- O ---- | EP1 |
                                        `-----'


T = 2 : EP1: routingcost = 40, hopcount = 3
        EP2: routingcost = 30, hopcount = 3

        - Route to EP1 has changed. Hopcount is now 3

        ==> EP2 is selected because routingcost is lower than for
        EP1, with the same hopcount value
                                          .-----.
              O ---------- O -------------| EP2 |
            /  \                          `-----'
          /     `-----.
        /              `------.         .-----.
Source ---------- X --------- [O] ---- | EP1 |
                                        `-----'
```

Figure 1: Endpoint re-selection using Multi-Cost ALTO request on
          updated cost values, upon a chnage in the route.

3.2.3.2. Case of a Multi-Cost ALTO query upon a cost value change

```
  T = 1 : EP1: routingcost = 30, hopcount = 2
           EP2: routingcost = 30, hopcount = 3
           ==> EP1 is selected because application is time-sensitive and
               hopcount metrics has higher weight

                                             .-----.
                O ---------- O ------------- | EP2 |
              /                              `-----'
            /
          /                               .-----.
        O ------------------------ O ---- | EP1 |
                                          `-----'


  T = 2 : EP1: routingcost = 40, hopcount = 2
          EP2: routingcost = 30, hopcount = 3
          Routingcost to EP1 has increased. Hopcount is the same.
          ==> Delay sensitive applications willing to minimize hopcount
              remain with EP1 while other applications may remain
              with EP2, that now has a lower routingcost.

                                            .-----.
                O ---------- O -------------| EP2 |
              /                             `-----'
            /
          /                               .-----.
        O ------------------------ O ---- | EP1 |
                                          `-----'
```

Figure 2: Endpoint selection using 2 Cost Types with joint request on updated cost values and for delay sensitive applications.

4. ALTO Protocol updates needed to support multi-cost transactions

To allow running Multi-Cost ALTO Services some minor changes in the base protocol are needed. A set of multi-cost specific media-types is introduced and the main updates consist into changing the JSON type of the value taken by a few members of the objects describing the information resources.

As written in the introduction, this section relies on the previous version of the ALTO protocol draft, see [ID-alto-protocol]. It partially integrates an update of the current version issued recently, see [ID-alto-protocol-11], that proposes a generic encoding of cost values in the 'JSONValue' data type. The proposed Multi-Cost specifications will be updated according to the outcome of WG discussions.

This section lists and details the proposed changes according to the previous ALTO protocol draft, [ID-alto-protocol] .

If members 'cost-type' and 'cost-mode' of objects InfoResourceCostMap, InfoResourceEndpointCostMap, ReqFilteredCostMap, ReqEndpointCostMap remain specified as single values in the base ALTO protocol, then Multi-Cost specific media types need to be used similarly to those specified in the previous version of this draft, see [draft-randriamasy-alto-multi-cost-05].

4.1. List of ALTO protocol updates required and recommended

The following updates to the current ALTO protocol, see [ID-alto-protocol], are required or recommended to support multi-cost ALTO transactions. The new resulting JSON formats are specified in the next sections.

o Updates required in the format of objects member(s):

   * Objects DstCosts (to destination PIDs) and EndpointDstCosts (to destination Endpoints): JSON type of cost value member evolves from JSONNumber to JSONArray.

   * Objects InfoResourceCostMap, ReqFilteredCostMap, ReqEndpointCostMap, InfoResourceEndpointCostMap: members 'cost-mode' and 'cost-type' have now an array of values rather than a single value.

o Updates recommended in the object structure:

   * Objects CostMapCapability and FilteredCostMapCapability: new member giving the maximum number of Cost Types in a response.

o Rules required on object member description:

   * Order in which the multiple cost values are provided in the responses,

   * Number of values in member 'cost-types' of objects InfoResourceCostMap, InfoResourceEndpointCostMap, ReqFilteredCostMap, ReqEndpointCostMap.

o Rule recommended on the cost value mode:

   * when the mode 'numerical' is available or applicable.

4.2. Updates required in the member format of objects

This section specifies the changes in the object member format that are required to enable multi-cost ALTO transactions.

The term Single Cost qualifies the items as they are specified in the current ALTO protocol draft, up to version 10

4.2.1. Cost value encoded in JSONArray

The fundamental change to support multi-cost is to encode the cost values with the type JSONArray. This way, the cost between 2 PIDs or to an Endpoint can be represented in a generic way:

- with several Cost Types,

- with Cost Types whose value can each be encoded with any type of JSON value.

For example, a multi-cost value represented with Cost Types (assuming they are supported by the ALTO Server):
["routingcost", "hopcount", "quarterlyvaluexxx", "statustring"]

will be encoded in the following JSON Array in a Multi Cost ALTO response:

```
[23, 6, [2, 5, 4, 1], "medium"]
```

The objects impacted by the encoding of ALTO Multi-Cost values in a JSONArray are: DstCosts and EndpointDstCosts. Full specification will be provided in later sections of this draft.

4.2.2. Format update on CostMode and CostType

In the base protocol, Objects InfoResourceCostMap, ReqFilteredCostMap, ReqEndpointCostMap, InfoResourceEndpointCostMap have members 'cost-mode' and 'cost-type' that list which Cost Type is reported and in which mode this Cost Type is represented.

In Multi-Cost ALTO several Cost Types are used per destination PID or Endpoint, so the member 'cost-type' of these objects must now be an array of values rather than a single value. Likewise, the member 'cost-mode' must now be an array, where each value reports the representation mode of the corresponding index in the 'cost-type' list.

The change on members 'cost-mode' and 'cost-type' from a single value to an array of values are specified in later sections.

4.3. Rules required on object member description

When several cost values are provided, it is necessary to unambiguously specify to which Cost Type each value corresponds and in which mode each value is provided.

4.3.1. Rule on cost value order in ALTO reponses

The cost values each Source/Destination pair MUST be provided in the same order as in the array of Cost Types. This way, the cost type values are provided without any ambiguity on the Cost Type they report on.

4.3.2. Rule on mapping for cost-type and cost-mode array members

The cost-mode array MUST be of the same size as the cost-type array. Each value of this array maps to the Cost Type ID at the same place in the Cost Type array and this value specifies the mode in which the value for this Cost Type is provided.

4.4. Updates recommended in the object structure

Objects MultiCostMapCapability and FilteredMultiCostMapCapability: new member giving the maximum number of Cost Types in a response.

4.5. Rule recommended on the cost value mode

In multi-cost transactions: when the mode 'numerical' is available for a Cost Type, it MUST be the one used to represent the cost values. In any case, the Cost Mode used for each Cost Type MUST be exactly specified.

The following example illustrates how how this rule can be applied:

```
Assume the Cost Types array:
   ["routingcost", "hopcount", "quarterlyvaluexxx", "statustring"]

The corresponding Cost Mode array should be (assuming that
these modes are supported):
   ["numerical", "numerical", "dynamic", "string"]

An example of values is:
   [23, 6, [21, 9, 4, 12], "medium"]
```

In this example, it is assumed that when the value of a Cost Type is expressed by an array of numbers such as [21, 9, 4, 12], the values in this array are expressed in the 'numerical' mode.

4.6. Extended constraints on multi-cost values

This draft proposes to extend the constraint tests in the base protocol to allow tests on the various costs in a request, and to allow more general predicates.

The base ALTO protocol allows optional contraints in the input parameters to a request for a Filtered Cost Map or the Endpoint Cost Service. The 'constraints' member is an array of expressions that all apply to the (single) requested Cost Type. The encoding of 'constraints' member, is fully specified in Section 6.8.2.2.3 "Input parameters" of the base protocol as follows:

A constraint contains two entities separated by whitespace:
(1) an operator,'gt' for greater than, 'lt' for less than,
'ge' for greater than or equal to, 'le' for less than or equal to,
or 'eq' for equal to
(2) a target cost value. The cost value is a number that MUST be
defined in the same units as the Cost Type indicated by the costtype
parameter
    ...
If multiple 'constraint' parameters are specified, they are
interpreted as being related to each other with a logical AND.

Such a specification covers multiple predicates on one metric such as:

     'routingcost' values belong to [6, 20)

However, an application

4.6.1. Use cases for mutli-cost multi-operator constraints

Suppose that an application uses information on the ALTO Cost Types 'hopcount' and 'routingcost'. This application may want to select paths or Endpoints with bounds on values for both 'hopcount' and 'routingcost'. For instance solutions meeting a constraint like:

    'hopcount' values in [6,20) OR 'routingcost' values in [100,200]

Moreover, this application may be ready to make compromises and to select paths or Endpoints by bounding their cost values according to

two options:

1. either solutions with moderate 'hopcount' and high 'routingcost', for instance: 'hopcount' values in [6,20] AND 'routingcost' values in [100,200],

2. or solutions with higher 'hopcount' and moderate 'routingcost', for instance: 'hopcount' values in [20,50] AND 'routingcost' values in [30,100].

4.6.2. Extended constraints in Multi-Cost ALTO

This draft proposes to support the two above mentioned use cases by extending the scope of constraints in two ways:

- allow the 'constraint' member to be applicable to multiple Cost Types,

- allow the multiple constraints to be related to each other by both logical AND and logical OR.

The two options would be covered by a logical expression like:

```
    [('hopcount' ge 6) AND ('hopcount' lt 20) AND
    ('routingcost' ge 100) AND ('routingcost' le 200)]
OR
    [('hopcount' ge 20) AND ('hopcount' le 50) AND
   ('routingcost' ge 30) AND ('routingcost' le 100)]
```

A simple encoding of multi-cost constraints for such expressions is specified in Section 5.3.3 of this draft, describing the input parameters to request for Filtered Cost Map. This specification is applicable to the EP Cost service as well.

5. Protocol extensions for multi-cost ALTO transactions

This section proposes extensions of the ALTO protocol to support Multi Cost ALTO Services or provide additional ALTO information. It integrates discussions on the ALTO mailing list.

If an ALTO client desires information on several Cost Types, then instead of placing as many requests as costs, it may request and receive all the desired Cost Types in one single transaction.

The ALTO server then, provided it supports the requested Cost Types, and provided it supports multi-cost ALTO transactions, sends one

single response where for each {source, destination} pair, the cost values are arranged in an array, where each component corresponds to a specified Cost Type. The correspondence between the components and the Cost Types is implicitly indicated in the ALTO response. Indeed, the values in the Cost values MUST be provided in the same order as in the array of cost types indicated in the response.

The following ALTO services have corresponding Multi-Cost extensions:

- Information Resources Directory: extended with multi-cost related URIs and associated capabilities.

- Cost Map Service: extended with the Multi-Cost Map Service,

- Cost Map Filtering Service: extended with the Multi-Cost Map Filtering Service,

- Endpoint Cost Lookup Service: extended with the Endpoint Multi-Cost Lookup Service.

5.1. Information Resources Directory

When the ALTO server supports the provision of information on multiple costs in a single transaction, the Information Resources Directory will list the corresponding resources. The media type remains the same as in the current ALTO protocol.

5.1.1. Example of Multi-Cost specific resources in the IRD

The following is an example Information Resource Directory returned by an ALTO Server and containing Multi-Cost specific services: the Multi-Cost Map Service, Filtered Multi-Cost Map and the Endpoint Multi-Cost Service. It is assumed that the IRD contains usual ALTO Services as described in the example IRD of the current ALTO protocol. In this example, the ALTO Server additionally provides Multi-Cost Services in a specific folder of "alto.example.com" called "multi". This folder contains the Multi-Cost Maps, Filtered Multi-Cost Maps as well as the Endpoint Multi-Cost Service.

In this example, the ALTO IRD exposes Multi-Cost capabilities on cosy types "routingcost", "hopcount", "pathoccupationcost", that can be combined in a request. The values on these metrics are provided in numerical mode. Values provided for cost-type string are in "string" mode.

```
GET /directory HTTP/1.1
   Host: alto.example.com
   Accept: application/alto-directory+json,application/alto-error+json
```

```
   HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-directory+json

   {
     "resources" : [
       {

         .....
         Usual ALTO "single-cost" Services as described in current ALTO
Protocol
         .....

       }, {
         "uri" : "http://alto.example.com/multi/costmap",
         "media-types" : ["application/alto-multicostmap+json"],
         "capabilities" : {
           "cost-types" : [ "routingcost", "hopcount" ],
           "cost-modes" : [ "numerical", "numerical" ]
         }
       }, {
         "uri" : "http://alto.example.com/multi/costmap/filtered",
         "media-types" : ["application/alto-multicostmap+json" ],
         "accepts" : ["application/alto-multicostmapfilter+json" ],
         "capabilities" : {
           "cost-constraints" : true,
           "max-cost-types" : 3,
           "cost-types" : [ "routingcost", "hopcount", "pathoccupationcost" ],
           "cost-modes" : [ "numerical", "numerical", "numerical" ]
         }
       }, {
         "uri" : "http://alto.example.com/multi/endpointmulticost/lookup",
         "media-types" : [ "application/alto-endpointmulticost+json" ],
         "accepts" : [ "application/alto-endpointmulticostparams+json" ],
         "capabilities" : {
           "cost-constraints" : true,
           "max-cost-types" : 2,
           "cost-types" : [ "routingcost", "hopcount", "status" ],
           "cost-modes" : [ "numerical", "numerical", "string" ]
         }
       }
     ]
   }
```

5.2. Multi-Cost Map Service

This section introduces a new media-type for the Multi-Cost map. For each source/destination pair of PIDs, it provides the values of the different Cost Types supported for the Multi-Cost map, in the same order as in the list of Cost Types specified in the capabilities.

A Multi-Cost Map MAY be provided by an ALTO Server.

Note that the capabilities specify implicitly the order in which the different Cost Type values will be listed in the Cost Map.

The Cost Type values in the responses are encoded as a JSONArray of cost values for the different Cost Types.

Note that values in a Multi-Cost map are arrays of values of the various Cost Types. If the ALTO server does not have the value for a particular Cost Type for a source/destination PID pair, the server MUST use 'null' (a reserved JSON symbol) for that location in the array. If the ALTO server does not have a value for any of the Cost Types for a given source/destination pair -- that is, the array is a list of nulls -- then the ALTO server MAY omit the entry for that source/destination pair.

5.2.1. Media Type

The media type is "application/alto-multicostmap+json".

5.2.2. HTTP Method

This resource is requested using the HTTP GET method.

5.2.3. Input Parameters

None.

5.2.4. Capabilities

This resource may be defined for multiple Cost Types and Cost Modes. The capabilities of an ALTO Server URI providing this resource are defined by a JSON Object of type CostMapCapability:

```
object {
     CostType cost-types<0..*>;
     CostMode cost-modes<0..*>;
   } MultiCostMapCapability;
```

with members

cost-types The Cost Types (Section 5.XX) supported by the corresponding URI. If not present, this member MUST be interpreted as an empty array.

cost-modes The Cost Modes (Section 5.XX) supported for each of the supported Cost Types listed in the array 'cost-types'. This array MUST have the same size as the array 'cost-types', and each member of this array MUST give the mode of the Cost Type at that index.

An ALTO Server MUST support all of the Cost Types listed here for each of the listed Cost Modes. Note that an ALTO Server may provide multiple Cost Map Information Resources, each with different capabilities.

An ALTO Server supporting the Multi-Cost Map service, MUST support the Cost mode 'numerical' for all supported Cost Types encoded with the 'JSONNumber' type.

5.2.5. Response

The returned InfoResourceEntity object has "data" member of type InfoResourceMultiCostMap:

```
   object DstMultiCosts {
     JSONArray [PIDName];
     ...
   };

   object {
     DstMultiCosts [PIDName]<0..*>;
     ...
   } MultiCostMapData;

   object {
     CostType    cost-type<0..*>;
     CostMode    cost-mode<0..*>;
     JSONString  map-vtag;
     MultiCostMapData map;
   } InfoResourceMultiCostMap;
```

with members:

cost-mode Array of Cost Modes (Section xxx) used in the Cost Map. This array MUST have the same size as the array 'cost-types'. where each member of the cost-mode array is the Cost Mode used for the Cost Type at the same place in the array.

cost-type The array of Cost Types (Section xxx) used in the Cost Map.

map-vtag The Version Tag (Section xx) of the Network Map used to generate the Cost Map.

map The Cost Map data itself.

MultiCostMapData is a JSON object with each member representing a single Source PID; the name for a member is the PIDName string identifying the corresponding Source PID. For each Source PID, a DstMultiCosts object denotes the associated costs to a set of destination PIDs each identified by a string indexed by PIDName. For each destination PID, object DstMultiCost[PIDName] provides an array of one or several values, each corresponding to the Cost Type listed at the same place in the 'cost-type' array. This array MUST have the same size as the 'cost-type' array. The values in the DstMultiCosts[PIDName] array MUST be listed in the same order as in the 'cost-type' array.

The returned Cost Map MUST include the required Path Costs for each pair of Source and Destination PID for which this information is available. If a cost value is not defined, it MUST be replaced by

the reserved JSON symbol 'null'.

The members 'cost-mode' and 'cost-type' MUST be arrays with the same number of elements.

5.2.6. Example

This example illustrates a 'static' multi-cost' ALTO transaction, where the utilized Cost Types all have 'static' values. We assume here that the Cost Types available at the ALTO Server are "routingcost" and "hopcount" and the 'numerical' mode is available for both of them. The "routingcost" may be based on monetary considerations where as the "hopcount" is used to report on the path delay. We also assume that ALTO server does not know the value of the "routingcost" between PID2 and PID3, and hence uses null for those costs.

```
GET /multicostmap/num HTTP/1.1
   Host: alto.example.com
   Accept: application/alto-multicostmap+json,application/alto-error+json

HTTP/1.1 200 OK
   Content-Length: [TODO]
   Content-Type: application/alto-multicostmap+json

   {
     "meta" : {},
     "data" : {
       "cost-mode" : ["numerical", "numerical"],
       "cost-type" : ["routingcost", "hopcount"],
       "map-vtag"  : "1266506139",
       "map" : {
         "PID1": { "PID1":[1,0],     "PID2":[5,23],   "PID3":[10,5] },
         "PID2": { "PID1":[null,5], "PID2":[1,0],     "PID3":[15,9] },
         "PID3": { "PID1":[20,12],  "PID2":[null,1], "PID3":[1,0]  }
       }
     }
   }
```

5.3. Filtered Multi-Cost Map

A Multi-Cost Map may be very large. In addition, an Application Client assisted by the ALTO Client does not necessarily need the Cost Types for all the source/destination PID pairs.

Therefore applications may more likely use Cost Map information filtered w.r.t. the Cost types as well as the source/destination

pairs of PIDs. This section specifies Filtered Multi-Cost Maps.

A Filtered Multi Cost Map is a Cost Map Information Resource for which an ALTO Client may supply additional parameters limiting the scope of the resulting Cost Map. A Filtered Multi Cost Map MAY be provided by an ALTO Server.

5.3.1. Media Type

The media type is "application/alto-multicostmap+json".

5.3.2. HTTP Method

This resource is requested using the HTTP POST method.

5.3.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies the input parameters with a data format indicated by the media type "application/alto-multicostmapfilter+json", which is a JSON Object of type ReqFilteredMultiCostMap, where:

```
 object {
   PIDName srcs<0..*>;
   PIDName dsts<0..*>;
 } PIDFilter;

 object {
   CostType    cost-type<0..*>;
   CostMode    cost-mode<0..*>;
   JSONString  constraints<0..*>;      [OPTIONAL]
   JSONArray   or-constraints<0..*>;   [OPTIONAL]
   PIDFilter   pids;                   [OPTIONAL]
 } ReqFilteredMultiCostMap;
```

with members:

cost-type The Cost Types for the returned costs.
Each listed Cost Type MUST be one of the supported Cost Types indicated in this resource's capabilities.

cost-mode The Cost Mode for each of the returned Cost Types.
As for the choice of Cost Modes, the ALTO Clients SHOULD be cognizant of operations applicable to different Cost Modes. For Cost types values encoded with the 'JSONNumber' type, the Cost Mode SHOULD be numerical when the purpose is to perform

multi-variate optimization.

constraints
Defines an array of additional constraints. The ALTO server MUST return the costs for all source/destination pair that satisfy all constraints in this list, and no other costs. This parameter MUST NOT be specified if this resource's capabilities (Section XXXX?) indicate that constraint support is not available. Each string in the 'constraint' array MUST contain three entities separated by whitespace, in the following format:

```
       [index] op value
```

'Index' is a number between 0 and the number of Cost Types minus 1, and indicates the Cost Type to which this constraint applies. (The square brackets ([]) surrounding 'index' are required syntactic sugar. They serve as a reminder that 'index' is an array index, not a value to test, and they avoid unusual-looking constraints such as "1 ge 5".) 'Op' is an operator: 'gt' for greater than, 'lt' for less than, 'ge' for greater than or equal to, 'le' for less than or equal to, 'eq' for equal to, or 'ne' for not equal to. 'Value' is a target cost value to compare against the indicated Cost Type. For numeric Cost Types, 'value' MUST be a number defined in the same units as the Cost Type indicated by 'index'. ALTO servers SHOULD use at least IEEE 754 doubleprecision floating point [IEEE.754.2008] to store the cost value, and SHOULD perform internal computations using double-precision floating-point arithmetic. For string Cost Types, 'value' MUST be a string enclosed in single quotes ('). For array-valued Cost Types, 'eq' is true iff one of the Cost Type values is equal to 'value', and 'ne' is true iff none of the Cost Type values are equal to 'value'. The other operators are not defined for array-valued Cost Types.

or-constraints
Defines an array of arrays of constraint strings. The individual constraint strings MUST be in the same format as strings in the 'constraints' parameter. The ALSO server MUST return costs that satisfy all constraints in one or more of the inner lists, and no other costs. That is, 'or-constraints' is the logical OR of ANDs. The client MUST NOT specify this parameter if this resource's capabilities (Section XXXX?) indicate that constraint support is not available. The client MUST NOT specify both a 'or-constraints' and a 'constraints' parameter.

pids A list of Source PIDs and a list of Destination PIDs for which Path Costs are to be returned. If a list is empty, the ALTO

Server MUST interpret it as the full set of currently-defined PIDs. The ALTO Server MUST interpret entries appearing in a list multiple times as if they appeared only once. If the "pids" member is not present, both lists MUST be interpreted by the ALTO Server as containing the full set of currently-defined PIDs.

5.3.4. Capabilities

The URI providing this resource supports all capabilities documented in Section 7.7.2.2.4 (with identical semantics), plus additional capabilities. In particular, the capabilities are defined by a JSON object of type FilteredMultiCostMapCapability:

```
object {
  CostMode   cost-modes<0..*>;
  CostType   cost-types<0..*>;
  JSONBool   cost-constraints;
  JSONNumber max-cost-types; [OPTIONAL]
} FilteredMultiCostMapCapability;
```

with members:

cost-modes See Section 4.2.5 of this MC draft

cost-types See Section 4.2.5 of this MC draft

max-cost-types Indicates the maximum number of cost values the ALTO Server can provide in a multi-cost array of a Multi-Cost Map.

cost-constraints If true, then the ALTO Server allows cost constraints to be included in requests to the corresponding URI. If not present, this member MUST be interpreted as if it specified false.

5.3.5. Response

See Section on Multi Cost Map Service of this draft for the format. The returned Cost Map MUST NOT contain any source/destination pair that was not indicated (implicitly or explicitly) in the input parameters. If the input parameters contain a PID name that is not currently defined by the ALTO Server, the ALTO Server MUST behave as if the PID did not appear in the input parameters. If any constraints are specified, Source/Destination pairs for which the

Path Costs do not meet the constraints MUST NOT be included in the returned Cost Map. If no constraints were specified, then all Path Costs are assumed to meet the constraints.

5.3.6. Example 1

```
POST multi/multicostmap/filtered HTTP/1.1
Host: alto.example.com
Content-Type: application/alto-multicostmapfilter+json
Accept: application/alto-multicostmap+json,application/alto-error+json

{
  "cost-mode" : ["numerical", "numerical"],
  "cost-type" : ["routingcost", "hopcount"],
  "pids" : {
    "srcs" : [ "PID1" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}


HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-multicostmap+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : ["numerical", "numerical"],
    "cost-type" : ["routingcost", "hopcount"],
    "map-vtag" : "1266506139",
    "map" : {
      "PID1": { "PID1": [1,6],  "PID2": [5,23],  "PID3": [10,5] }
    }
  }
}
```

5.3.7. Example 2

This is an example of using constraints to restrict returned source/ destination PID pairs to those with 'routingcost' between 5 and 10, or 'hopcount' equal to 0.

```
POST multi/multicostmap/filtered HTTP/1.1
Host: alto.example.com
Content-Type: application/alto-multicostmapfilter+json
Accept: application/alto-multicostmap+json,application/alto-error+json

{
  "cost-mode" : ["numerical", "numerical"],
  "cost-type" : ["routingcost", "hopcount"],
  "or-constraints" : [ ["[0] ge 5", "[0] le 10"],
                       ["[1] eq 0"] ]
  "pids" : {
    "srcs" : [ "PID1", "PID2" ],
    "dsts" : [ "PID1", "PID2", "PID3" ]
  }
}


HTTP/1.1 200 OK
Content-Length: [TODO]
Content-Type: application/alto-multicostmap+json

{
  "meta" : {},
  "data" : {
    "cost-mode" : ["numerical", "numerical"],
    "cost-type" : ["routingcost", "hopcount"],
    "map-vtag" : "1266506139",
    "map" : {
      "PID1": { "PID2": [5,23], "PID3": [10,5] }
      "PID2": { "PID2": [1,0]                  },
    }
  }
}
```

5.4. Endpoint Multi-Cost Service

The Endpoint Multi-Cost Service provides information on several Cost Types between individual Endpoints.

This service MAY be provided by an ALTO Server. It is important to note that although this resource allows an ALTO Server to reveal costs between individual endpoints, an ALTO Server is not required to do so. A simple alternative would be to compute the cost between two endpoints as the costs between the PIDs corresponding to the endpoints if these values are available for the requested Cost Types.

When the cost values are requested to perform multi-variate numerical

optimization and are each available in the 'numerical' mode, then the ALTO Client SHOULD request the 'numerical' mode in order to get a reliable result. Note that this consideration is outside the scope of the ALTO protocol as it relates to the responsibility of the ALTO Client and related entries. However common sense lead to warn that a necessary condition for vector ranking method to be reliable is that the components of the processed vectors are numerical and not ordinal values.

5.4.1. Media Type

The media type is "application/alto-endpointmulticost+json".

5.4.2. HTTP Method

This resource is requested using the HTTP POST method

5.4.3. Input Parameters

Input parameters are supplied in the entity body of the POST request. This document specifies input parameters with a data format indicated by media type "application/alto-endpointmulticostparams+json", which is a JSON Object of type ReqEndpointMultiCostMap:

```
object {
     TypedEndpointAddr srcs<0..*>; [OPTIONAL]
     TypedEndpointAddr dsts<1..*>;
} EndpointFilter;

object{
    CostType    cost-type<0..*>;
    CostMode    cost-mode<0..*>;
    JSONString  constraints<0..*>;      [OPTIONAL]
    JSONArray   or-constraints<0..*>;   [OPTIONAL]
    EndpointFilter endpoints;
} ReqEndpointMultiCostMap;
```

with members:

cost-mode Defined equivalently to the "cost-mode" input parameter of a Filtered Multi Cost Map.

cost-type Defined equivalently to the "cost-type" input parameter of a Filtered Multi Cost Map.

constraints Defined equivalently to the "constraints" input parameter of a Filtered Multi Cost Map.

or-constraints Defined equivalently to the "or-constraints" input parameter of a Filtered Multi Cost Map.

endpoints A list of Source Endpoints and Destination Endpoints for which Path multiple Costs are to be returned. If the list of Source Endpoints is empty (or not included), the ALTO Server MUST interpret it as if it contained the Endpoint Address corresponding to the client IP address from the incoming connection (see Section 10.3 for discussion and considerations regarding this mode). The list of destination Endpoints MUST NOT be empty. The ALTO Server MUST interpret entries appearing multiple times in a list as if they appeared only once.

5.4.4. Capabilities

See section on Filtered Multi Cost Map capabities in this draft.

5.4.5. Response

The returned InfoResourceEntity object has "data" member equal to InfoResourceEndpointMultiCostMap, where:

```
      object EndpointDstMultiCosts {
        JSONArray [TypedEndpointAddr];
        ...
      };

      object {
        EndpointDstMultiCosts [TypedEndpointAddr]<0..*>;
        ...
      } EndpointMultiCostMapData;

      object {
        CostMode            cost-mode<0..*>;
        CostType            cost-type<0..*>;
        EndpointMultiCostMapData map;
      } InfoResourceEndpointMultiCostMap;
```

InfoResourceEndpointMultiCostMap has members:

cost-type<0..*> The Cost Types used in the returned Cost Map.

cost-mode<0..*> The Cost Mode for each of the Cost Types used in the returned Cost Map.

map The Endpoint Multi-Cost Map data itself.

EndpointMultiCostMapData is a JSON object with each member representing a single Source Endpoint specified in the input parameters; the name for a member is the TypedEndpointAddr string identifying the corresponding Source Endpoint. For each Source Endpoint, a EndpointDstMultiCosts object denotes the cost vector associated to each Destination Endpoint specified in the input parameters; the name for each member in the object is the TypedEndpointAddr string identifying the corresponding Destination Endpoint.

5.4.6. Example

```
  POST multi/endpointmulticost/lookup HTTP/1.1
  Host: alto.example.com
  Content-Length: [TODO]
  Content-Type: application/alto-endpointmulticostparams+json
  Accept: application/alto-endpointmulticost+json,application/alto-error+json

  {
    "cost-type" : ["routingcost", "hopcount"],
    "cost-mode" : ["numerical", "numerical"],
    "endpoints" : {
      "srcs": [ "ipv4:192.0.2.2" ],
      "dsts": [
        "ipv4:192.0.2.89",
        "ipv4:198.51.100.34",
        "ipv4:203.0.113.45"
      ]
    }
  }


  HTTP/1.1 200 OK
  Content-Length: [TODO]
  Content-Type: application/alto-endpointmulticost+json

  {
    "meta" : {},
    "data" : {
      "cost-type" : ["routingcost", "hopcount"],
      "cost-mode" : ["numerical", "numerical"],
      "map" : {
        "ipv4:192.0.2.2": {
          "ipv4:192.0.2.89"    : [1, 7],
          "ipv4:198.51.100.34" : [2, 4],
          "ipv4:203.0.113.45"  : [3, 2]
        }
      }
    }
  }
```

6. IANA Considerations

Information for the ALTO Endpoint property registry maintained by the IANA and related to the new Endpoints supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Endpoint Property Registry" of

[ID-alto-protocol],

Information for the ALTO Cost Type Registry maintained by the IANA and related to the new Cost Types supported by the acting ALTO server. These definitions will be formulated according to the syntax defined in Section on "ALTO Cost Type Registry" of [ID-alto-protocol],

6.1. Information for IANA on proposed Cost Types

When a new ALTO Cost Type is defined, accepted by the ALTO working group and requests for IANA registration MUST include the following information, detailed in Section 11.2: Identifier, Intended Semantics, Security Considerations.

6.2. Information for IANA on proposed Endpoint Propeeries

Likewise, an ALTO Endpoint Property Registry could serve the same purposes as the ALTO Cost Type registry. Application to IANA registration for Endpoint Properties would follow a similar process.

7. Acknowledgements

The authors would like to thank Dave Mac Dysan and Vijay Gurbani for fruitful discussions and comments on this draft and previous versions.

Sabine Randriamasy is partially supported by the MEDIEVAL project (http://www.ict-medieval.eu/), a research project supported by the European Commission under its 7th Framework Program (contract no. 248565). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the MEDIEVAL project or the European Commission.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5693] "Application Layer Traffic Optimization (ALTO) Problem Statement", October 2009.

8.2. Informative References

[ID-ALTO-Requirements13]
"draft-ietf-alto-reqs-13.txt", January 2012.

[ID-alto-protocol]
, Eds., ""ALTO Protocol" draft-ietf-alto-protocol-10.txt", October 2011.

[ID-alto-protocol-11]
, Eds., ""ALTO Protocol" draft-ietf-alto-protocol-11.txt", March 2012.

[draft-jenkins-alto-cdn-use-cases-01]
""Use Cases for ALTO within CDNs" draft-jenkins-alto-cdn-use-cases-01", June 2011.

[draft-randriamasy-alto-cost-schedule-01]
"ALTO Cost Schedule", July 2012.

[draft-randriamasy-alto-multi-cost-05]
"Multi-Cost ALTO", October 2011.

Authors' Addresses

Sabine Randriamasy (editor)
Alcatel-Lucent Bell Labs
Route de Villejust
NOZAY 91460
FRANCE

Email: Sabine.Randriamasy@alcatel-lucent.com

W. D. Roome
Alcatel-Lucent Bell Labs
600 Mountain Ave.
Murray Hill, NJ 07974
USA

Phone:
Fax:
Email: W.Roome@alcatel-lucent.com
URI:

Nico Schwan

Email: ietf@nico-schwan.de