



**Implementation Agreement for Dynamic Policy
Control using COPS-PR**

MSF-IA-COPS-PR.001-FINAL

MultiService Forum Implementation Agreement

Contribution Number: **msf2004.222.05**
Last Saved: **07/18/2005 16:07 PM**
Working Group: **Protocol and Control**
Title: **IA for dynamic policy control using COPS-PR**

Source:

Yasuyuki Matsuoka	Susumu Yamamoto	Jun Nishikido
NTT	NTT	NTT
matsuoka.yasuyuki@lab.ntt.co.jp	yamamoto.susumu@lab.ntt.co.jp	nishikido.jun@lab.ntt.co.jp

Abstract:

This contribution is a draft Implementation Agreement of COPS Usage Policy Provisioning (COPS-PR) for dynamic QoS policy controls in edge nodes as an IF-3, which is the protocol between edge nodes and the bandwidth manager. The target is to set up QoS policy for each microflow on the edge nodes forming a DiffServ core network.

The goal of the MSF is to promote multi-vendor interoperability as part of a drive to accelerate the deployment of next generation networks. To this end the MSF looks to adopt pragmatic solutions in order to maximize the chances for early deployment in real world networks.

To date the MSF has defined a number of detailed Implementation Agreements and detailed Test Plans for the signaling protocols between network components and is developing additional Implementation Agreements and Test Plans addressing some of the other technical issues such as QoS and Security to assist vendors and operators in deploying interoperable solutions.

The MSF welcomes feedback and comment and would encourage interested parties to get involved in this work program. Information about the MSF and membership options can be found on the MSF website <http://www.msforum.org/>

DISCLAIMER

The information in this publication is believed to be accurate as of its publication date. Such information is subject to change without notice and the MultiService Forum is not responsible for any errors or omissions. The MultiService Forum does not assume any responsibility to update or correct any information in this publication. Notwithstanding anything to the contrary, neither the MultiService Forum nor the publisher make any representation or warranty, expressed or implied, concerning the completeness, accuracy, or applicability of any information contained in this publication. No liability of any kind whether based on theories of tort, contract, strict liability or otherwise, shall be assumed or incurred by the MultiService Forum, its member companies, or the publisher as a

result of reliance or use by any party upon any information contained in this publication. All liability for any implied or express warranty of merchantability or fitness for a particular purpose is hereby disclaimed.

The receipt or any use of this document or its contents does not in any way create by implication or otherwise:

Any express or implied license or right to or under any MultiService Forum member company's patent, copyright, trademark or trade secret rights which are or may be associated with the ideas, techniques, concepts or expressions contained herein; nor

Any warranty or representation that any MultiService Forum member companies will announce any product(s) and/or service(s) related thereto, or if such announcements are made, that such announced product(s) and/or service(s) embody any or all of the ideas, technologies, or concepts contained herein; nor

Any commitment by a MultiService Forum company to purchase or otherwise procure any product(s) and/or service(s) that embody any or all of the ideas, technologies, or concepts contained herein; nor

Any form of relationship between any MultiService Forum member companies and the recipient or user of this document.

Implementation or use of specific MultiService Forum Implementation Agreements, Architectural Frameworks or recommendations and MultiService Forum specifications will be voluntary, and no company shall agree or be obliged to implement them by virtue of participation in the MultiService Forum.

For addition information contact:

MultiService Forum
39355 California Street, Suite 307
Fremont, CA 94538
USA

Phone: +1 510 608-5922

Fax: +1 510 608-5917

info@msforum.org

<http://www.msforum.org>

1 MultiService Forum

The goal of the MSF is to promote multi-vendor interoperability as part of a drive to accelerate the deployment of next generation networks. To this end the MSF looks to adopt pragmatic solutions in order to maximize the chances for early interoperability in real world networks.

To date the MSF has defined a number of detailed Implementation Agreements and detailed Test Plans for the signaling protocols between network components and are developing additional Implementation Agreements and Test Plans addressing some of the other technical issues such as QoS and Security to assist vendors and operators in deploying interoperable solutions.

In 2002, the MSF held a “Global MSF Interoperability 2002” (GMI 2002) event that tested interoperability between next generation network elements situated in Asia, Europe and North America. GMI 2002 validated the MSF release 1 architectural framework and Implementation Agreements by subjecting them to interoperability testing based on realistic network scenarios.

The MSF welcomes feedback and comment and would encourage interested parties to get involved in this work program. Information about the MSF and membership options can be found on the MSF website <http://www.msforum.org/>

2 Scope of this Document

This document defines an implementation agreement for Common Open Policy Service Protocol usage for Policy Provisioning (COPS-PR) [3] for IF-3 in the MSF QoS architecture [1]. COPS-PR is the protocol to activate QoS policies from the Policy Decision Point (PDP) to Policy Enforcement Point (PEP) based on the COPS provisioning architecture. The target for using COPS-PR as IF-3 is to set up a QoS policy for each microflow at the edge nodes forming a DiffServ core network.

This draft supposes that the signal from a SIP server is the trigger to push a set of QoS policies carried by COPS-PR from the Bandwidth Manager (BM) to edge nodes. This set has the named data structure, known as a Policy Information Base (PIB) which is specified in Framework PIB [4]. To provide an MSF QoS architecture [1] in a DiffServ core network, the standard DiffServ PIB [3] is described for QoS functions in an edge node, such as packet classifier, packet filter, rate meter, packet dropper, queues, packet scheduler, marker, and shaper [6].

Since there is no standard PIB capable of performing MPLS policies at this point, MPLS capability in the core network is out of the scope in this IA.

When a core network is based on hop-by-hop routing, this IA covers the required QoS policy control on a per-microflow basis in the MSF QoS architecture.

There are three protocols in the table in MSF besides COPS-PR: they are ETSI H.248 EMP [7], GCP-M [8], and Pinhole Control Protocol [9]. However, these three protocols are included in the MSF original specification, while this IA just profiles IETF standards to provide the DiffServ core network.

The emergency or priority calls such as GETS, 911 and 999 would be specified for further study before the MSF architecture or service discussion about these calls whether would be exempt from dynamic QoS policy controls or not.

3 COPS Provisioning Architecture

Two function blocks (PDP and PEP) are determined in the COPS provisioning architecture (Figure 3-1). Triggered by certain external events, PDP is responsible for deciding to push appropriate policies onto PEP.

PEP provides QoS control packet-by-packet based on policies from PDP. The MSF QoS architecture conforms to the COPS provisioning architecture in terms of the PDP and PEP function model. The Bandwidth Manager (BM) has a PDP function inside, and edge nodes have PEP. (Figure 3-2)

Since a session establishment request from a Call Agent or SIP proxy triggers a policy decision in BM in the case of the MSF QoS architecture, COPS-PR conforms to the architecture. COPS-PR is triggered to push QoS policies by external events in PDP, while COPS-RSVP is triggered by events in PEP.

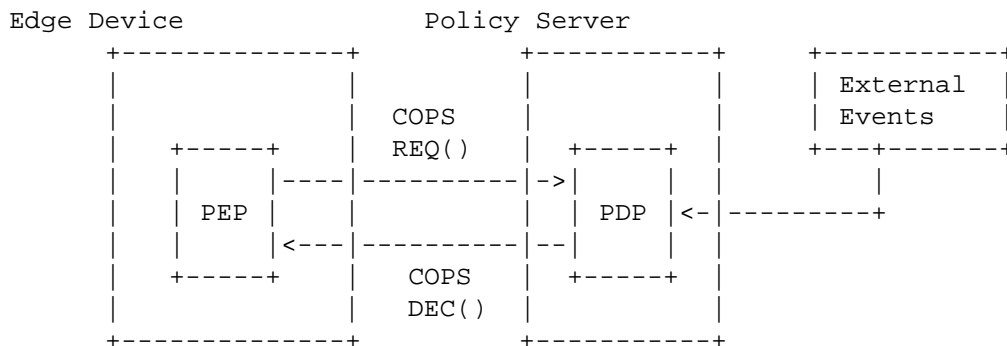


Figure 3-1: COPS provisioning architecture.

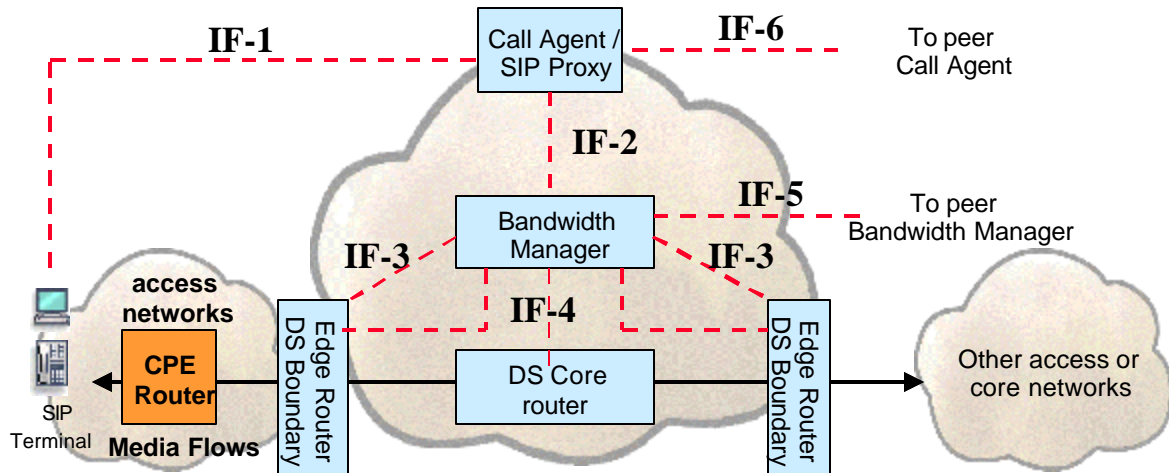


Figure 3-2: MSF architecture.

4 COPS-PR Protocol Stack

4.1 Protocol stack

Figure 4-1 shows COPS-PR protocol stacks.

For this profile, TCP transport shall be used on port 3288 and IPv4 addresses is mandatory. IPv6 address is optional.

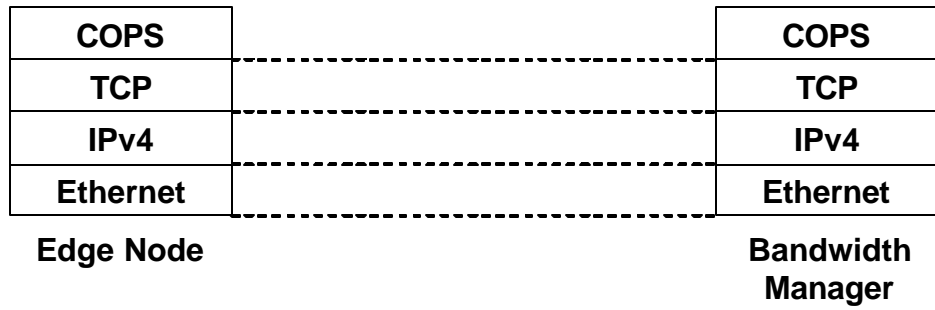


Figure 4-1: Protocol stack (pure IP).

5 COPS-PR Header

For this profile, the following fields are used in common header of COPS-PR messages.

5.1 Common header

Refer to Appendix C for each field.

Name	Description	Mandatory/optional
Version	Protocol Version = 1	Mandatory
Op-Code	1 = Request (REQ)	Mandatory
	2 = Decision (DEC)	Mandatory
	3 = Report State (RPT)	Mandatory
	4 = Delete Request State (DRQ)	Mandatory
	5 = Synchronize State Request (SSQ)	Mandatory
	6 = Client-Open (OPN)	Mandatory
	7 = Client-Accept (CAT)	Mandatory
	8 = Client-Close (CC)	Mandatory
	9 = Keep-Alive (KA)	Mandatory
	10= Synchronize Complete (SSC)	Mandatory
C-Num	0x0 Unsolicited Message Flag Bit: unsolicited 0x1 Solicited Message Flag Bit: solicited	Mandatory
Client-type	0 = authentication process in OPN message	Optional for PEPs to send Mandatory for PDPs to receive
	2 = DiffServ PIB	Mandatory
	0x4000 - 0x7FFF = private use	Mandatory
	0x8000 - 0xFFFF = enterprise use managed by IANA	Mandatory
Message Length	Size of message in octets, which includes the standard COPS header and all encapsulated objects.	Mandatory

6 COPS-PR Object Formats

For this profile, the following fields and codes are used in objects of COPS-PR messages.
Refer to Appendix C for each field.

6.1 Specific object format

Name	Description	Mandatory/optional		
Length	Two-octet value	Mandatory		
C-Num, C-Type	1 = Handle	1	Mandatory	
	2 = Context	1	Mandatory	
	3 = In Interface	1	Not available for COPS-PR	
	4 = Out Interface	1	Not available for COPS-PR	
	5 = Reason code	1	Mandatory	
	6 = Decision	1 = Decision Flags	1	Mandatory
		2 = Stateless Data		Not available for COPS-PR
		3 = Replacement Data		Not available for COPS-PR
		4 = Client Specific Decision Data		Not available for COPS-PR
		5 = Named Decision Data		Mandatory
	7 = LPDP Decision	1	Not available for COPS-PR	
	8 = Error	1	Mandatory	
	9 = Client Specific Info	1 = Signaled ClientSI		Not available for COPS-PR
		2 = Named ClientSI		Mandatory
	10 = Keep-Alive Timer	1	Mandatory	
	11 = PEP Identification	1	Mandatory	
12 = Report Type	1	Mandatory		
13 = PDP Redirect Address	1 = IPv4 address		Optional for PEPs to send Mandatory for PDPs to receive	
	2 = IPv6 address		Optional for PEPs to send Mandatory for PDPs to receive	
14 = Last PDP Address	1 = IPv4 address		Optional for PEPs to send Mandatory for PDPs to receive	
	2 = IPv6 address		Optional for PEPs to send Mandatory for PDPs to receive	
15 = Accounting Timer	1	Optional for PEPs to send Mandatory for PDPs to receive		
16 = Message Integrity	1	Optional for PEPs to send Mandatory for PDPs to receive		
(Object contents)		Mandatory		

6.1.1 Handle object

"Client Handle" is an identifier of a combination of policies to be set or modified or deleted on PEP as a cluster, and is used to share policy status between PDP and PEP. These handles are created as unique identifiers on the PEP and notified to PDP with a REQ message after COPS session is established.

When PDP intends to create or modify or delete a policy set, PDP issues a DEC message with corresponding "Client Handle". And in the case of changing the status of a policy set, PEP reports the change on PDP with the "Client Handle". This field is defined in Appendix A.

RFC only describes “Client Handle” is used to manage request state between PEPs and PDPs. In this document, the assignment of “Client Handle” is defined according to real service as following ways. There are two ways how policies can be managed between the PEP and PDP; one is non-shared (dedicated) policy, or the other is shared policy as a group. These ways depend on how the policies are associated with client handles. The latter one includes three types of client handles according to how to share polices on the edge routes. Every ways of client handles are used to provide dynamic policy control on the edge router per media session.

6.1.1.1 Dedicated policy for Client Handle

Each policy group is associated with each client handle. Those policy groups are handled as different groups even if there are the several definite same contents. Accordingly, these client handles are assigned to flows or subscribers on sub-interface or VLANs respectively.

A PDP can control the policy of the subscriber using the corresponding client handle. This client handle includes dedicated policy for the corresponding interface.

6.1.1.2 Shared policy among Client Handles

(1) Client handles per service:

The policy group is associated with the common shared client handle. Several specific flows and subscribers belonging to the same service group should be assigned to the same shared client handles. These subscribers are categorized some SUBSCRIBER TYPES. “Client Handle” is assigned per SUBSCRIBER TYPES. Therefore, PDP can manage policies state in a certain degree of service group. This method takes advantage of the high granularity of operation however there might be scalability issues.

(2) Client handles with a shared handle and interfaces handles

There is another example for the special common handle that is utilized by the PDP to install or remove policy configuration not directly associated with a specific handle, request state and an interface. These policies are associated with one common client handle on whole policies on the PEP . Other policies which are independently provisioned would be attached to specific interfaces or sub-interfaces via the associated unique request state handles. These interface handles refer to the common client handle for policies to activate them on the attached interface.

(3) A shared client handle per COPS session

In this way, all of policies related to some media session on PDP using one client handle on this COPS-PR session between PDP and PEP. Whenever the PDP changes any policies on the only one subscriber, this client handle is used to convey the difference of configuration. This method takes advantage of higher scalability however the granularity of operation is lower than other shared policies methods.

PEPs should support dedicated policy for Client Handle and shall support shared policy among Client Handles. PDPs should support both of ways.

6.1.2 Context object (context)

R-Type (Request type flag)

Bits	Description	Mandatory/optional
0x01	Incoming-message/admission control request	FFS
0x02	Resource-allocation request	FFS
0x04	Outgoing-message request	FFS
0x08	Configuration request	Mandatory

M-Type (Message type)

Client-specific 16-bit values of protocol message types

6.1.3 Reason object (reason)

Reason Code:

Code	Description	Mandatory/optional
1	Unspecified	Optional for PEPs to send Mandatory for PDPs to receive
2	Management	FFS
3	Preempted (Another request state takes precedence)	Optional for PEPs to send Mandatory for PDPs to receive
4	Tear (Used to communicate a signaled state removal)	FFS
5	Timeout (Local state has timed-out)	Optional for PEPs to send Mandatory for PDPs to receive
6	Route change (Change invalidates request state)	Mandatory
7	Insufficient resources (No local resource available)	Mandatory
8	BM's directive (BM decision caused the delete)	Optional for PEPs to send Mandatory for PDPs to receive
9	Unsupported decision (BM decision not supported)	Optional for PEPs to send Mandatory for PDPs to receive
10	Synchronize handle unknown	FFS
11	Transient handle (stateless event)	FFS
12	Malformed decision (could not recover)	Optional for PEPs to send Mandatory for PDPs to receive
13	Unknown COPS Object from BM: Sub-code (octet 2) contains unknown object's C-Num and (octet 3) contains unknown object's C-Type.	Optional for PEPs to send Mandatory for PDPs to receive

6.1.4 Decision object (decision)

Decision Flags (mandatory)

Name	Description	Mandatory/Optional
Command-code	0 = NULL Decision (No configuration data available) 1 = Install (Admit request/Install configuration) 2 = Remove (Remove request/Remove configuration)	Mandatory
Flags	0x01 = Trigger error (Trigger error message if set) 0x02 = Request-state flag	Mandatory

6.1.5 Error object (Error)

Error-Code:

Code	Cause	Mandatory/optional
1	Bad handle	Optional for PEPs/PDPs to send Mandatory for PEPs/PDPs to receive
2	Invalid handle reference	FFS
3	Bad message format (Malformed Message)	FFS
4	Unable to process (server gives up on query)	FFS
5	Mandatory client-specific info missing	FFS
6	Unsupported client-type	FFS

7	Mandatory COPS object missing	FFS
8	Client Failure	Optional for PEPs/PDPs to send Mandatory for PEPs/PDPs to receive
9	Communication Failure	FFS
10	Unspecified	FFS
11	Shutting down	FFS
12	Redirect to Preferred Server	FFS
13	Unknown COPS Object: Sub-code (octet 2) contains unknown object's C-Num and (octet 3) contains unknown object's C-Type.	Optional for PEPs/PDPs to send Mandatory for PEPs/PDPs to receive
14	Authentication Failure	Optional for PEPs/PDPs to send Mandatory for PEPs/PDPs to receive
15	Authentication Required	Optional for PEPs/PDPs to send Mandatory for PEPs/PDPs to receive

6.1.6 Report-Type object (Report-Type)

Report-Type:

Code	Description	Mandatory/optional
1	Success : Decision was successful at the edge node	Mandatory
2	Failure : Decision could not be completed by edge node	Mandatory
3	Accounting: Accounting update for an installed state	Optional for PEPs to send Mandatory for PDPs to receive

7 COPS-PR Policy Provisioning object Formats

For this profile, the following fields and codes are used in objects of COPS-PR messages.
Refer to Appendix C for each field.

7.1 Specific sub-object format

Name	Description	Mandatory/optional
Length	Two-octet value	Mandatory
S-Num	1 = Complete PRID, Length = variable.	Mandatory
	2 = Prefix PRID, Length = variable.	Mandatory
	3 = EPD, Length = variable.	Mandatory
	4 = GPERR, Length = 8.	Mandatory
	5 = CPERR, Length = 8.	Mandatory
	6 = ErrorPRID, Length = variable.	Mandatory
S-Type	1 = BER (for BER)	Mandatory
(Sub-Object contents)		Mandatory

7.2 Global Provisioning Error object (GPERR)

Error-Code:

Code	Description	Mandatory/optional
availMemLow(1)		FFS
availMemExhausted(2)		FFS
unknownASN.1Tag(3)	The erroneous tag type SHOULD be specified in the Error Sub-Code field.	FFS
maxMsgSizeExceeded(4)	COPS message (transaction) was too big.	FFS
unknownError(5)		FFS
maxRequestStatesOpen(6)	No more Request-States can be created by the PEP (in response to a DEC message attempting to open a new Request-State).	FFS
invalidASN.1Length(7)	The ASN.1 object length was incorrect.	FFS
invalidObjectPad(8)	Object was not properly padded.	FFS
unknownPIBData(9)	Some of the data supplied by the PDP is unknown/unsupported by the PEP (but otherwise formatted correctly). PRC-specific error codes are to be used to provide more information.	FFS
unknownCOPSPROject(10)	Sub-code (octet 2) contains unknown object's S-Num and (octet 3) contains unknown object's S-Type.	FFS
malformedDecision(11)	Decision could not be parsed.	FFS

7.3 PRC Class Provisioning Error object (CPERR)

Error-Code:

Code	Description	Mandatory/optional
priSpaceExhausted(1)	No more instances may currently be installed in the given class.	Optional for PEPs to send Mandatory for PDPs to receive
priInstanceInvalid(2)	The specified class instance is currently invalid prohibiting	Optional for PEPs to send Mandatory for PDPs to receive
attrValueInvalid(3)	The specified value for identified attribute is illegal. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.	Optional for PEPs to send Mandatory for PDPs to receive
attrValueSupLimited(4)	The specified value for the identified attribute is legal but not currently supported by the device. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.	FFS
attrEnumSupLimited(5)	The specified enumeration for the identified attribute is legal but not currently supported by the device. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.	FFS
attrMaxLengthExceeded(6)	The overall length of the specified value for the identified attribute exceeds device limitations. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.	FFS
attrReferenceUnknown(7)	The class instance specified by the policy instance identifier does not exist. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.	Optional for PEPs to send Mandatory for PDPs to receive
priNotifyOnly(8)	The class is currently only supported for use by request or report messages prohibiting decision installation.	FFS
unknownPrc(9)	Attempt to install a PRI of a class not supported by PEP.	Optional for PEPs to send Mandatory for PDPs to receive
tooFewAttrs(10)	Received PRI has fewer attributes than required.	FFS
invalidAttrType(11)	Received PRI has an attribute of the wrong type.	Optional for PEPs to send Mandatory for PDPs to receive
deletedInRef(12)	Deleted PRI is still referenced by other (non) deleted PRIs.	FFS
priSpecificError(13)	The Error Sub-code field contains the PRC specific error code.	FFS

8 Timer

8.1 Maximum Keep Alive Timer Interval

“Maximum Keep Alive Timer Interval” is reported to edge nodes via a CAT message from BM. If Keep Alive messages are not reported while this timer is working, then the COPS session between BM and the edge node is considered to have failed, and the COPS session and TCP session will be disconnected. The interval time when the edge node sends out Keep Alive messages is set to one fourth of the “Maximum Keep Alive Timer Interval”.

When a time-out occurs, the entity that recognizes the failure of these messages may send a Client-Close (CC) message to the peering entity and disconnect the TCP session.

8.2 Long timer

The failure of the protocol should be detected by COPS-PR, not by TCP, for quick recovery. Therefore, a response timer should be implemented with the application level of COPS-PR. This timer is named long timer because this timer is longer than TCP retransmission timer.

The messages that need to receive responses are OPN, REQ, DEC, and SSQ. The timer starts when a message is sent and stops when a corresponding response message or KA message is received. If an entity does not receive any response after this timer times out, the entity may start action to disconnect COPS session with a Client- Close (CC) message, to delete corresponding client handle with a COPS Delete Request State (DRQ) message or to persuade PEP to send DRQ by sending a Decision (DEC) message with flag 0x02 on a Decision Flags Object.

The messages and expected responses are as follows:

	Corresponding messages	The action after time-out
1	OPN (the edge node) → CAT (BM)	• PEP sends <CC>
2	REQ (the edge node) → DEC (BM)	• PEP retries COPS session establishment with new client Handle.
3	DEC (BM) → RPT (the edge node)	• PDP sends <DEC> with 0x02 flag and deletes the corresponding policy state.
4	SSQ (BM) → SSC(the edge node)	• PDP sends <CC> when router restarts. • PDP may notify the timeout to operators when operators input synchronize command.

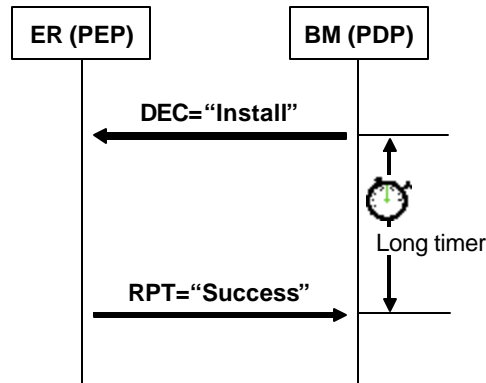


Figure 8-1: Long timer.

9 PIB (Policy Information Base)

The policy provisioning should be carried out using a policy information base (PIB) [4] on Client-specific Decision Data objects of a DEC message from BM. In this document, the PIBs defined in [4] and [5] are used for provisioning. (see Appendix A.)

10 Naming Policy

This section shows the default behavior for each event to provide dynamic QoS policy controls. See Appendix F. for details of the call flow.

10.1 Client Handles

The length of Client handle for dedicated and shared policy may be 4 octets basically (because the length of Client handle object is variable). A client handle's value increases sequentially and uniquely on PEP. PEP and PDP would retain policy states associated with these client handles.

When there is some policy state failure on the client handle, PEP recreates a new client handle. Therefore, PEP should define numbering areas for client handles; primary assignment area and secondary area etc.

Example: Handle area; 0x00000000 is for primary, 0x00010000 is for secondary. If a new subscriber is added, the handle value increases.

Unit	Client handle
Subscriber 1	0x00000001
Subscriber 2	0x00000002
--	--

----->
--(Policy state failure & reconnected)-->
-----{subscriber added}----->

Unit	Client handle
Subscriber 1	0x00000001
Subscriber 2	0x00010002
Subscriber 3	0x00000003

10.2 PEP ID

For this profile, a loopback address on PEP shall be used as PEPID.

11 Security

The security consideration is needed to avoid from abuse. Implementation of COPS-PR for MSF shall deal with the following authentication and security functions, and also the lower layer can achieve encrypted compunction for security.

However, this implementation would depend on management network architecture which is not further discussed in this IA. Therefore, authentication of COPS-PR layer is being as optional. We suppose that the management network architecture in this document would be closed away from user networks, or the outbound control model would be considered for the first interoperability testing to simplify the condition to test the implementations within.

11.1 Authentication (Optional)

Key ID and Sequence Number should be included in Integrity Object.

- Authentication by Key ID: prevention of fake BMs
- Sequence Number: prevention of replay attacks
 - This number should be initiated using random number selection and setup in OPN and CAT messages respectively including sequence number field in of Integrity object.

11.2 Encrypted Communication using (For further study)

COPS-PR would achieve the following two encrypted communication technologies:

- IPSec
- TLS (SSL)

These protocols also have authentication capability. The interaction or overlap between these mechanisms and the COPS-PR Integrity Object are for further study.

12 References

- [1] MSF-AF-QOS.001-FINAL-Quality of Service for next generation VoIP networks solution framework.
- [2] The COPS (Common Open Policy Service) Protocol: RFC2748
- [3] COPS Usage for Policy Provisioning (COPS-PR): RFC3084
- [4] Framework Policy Information Base: RFC3318
- [5] Differentiated Services Quality of Service Policy Information Base: RFC3317
- [6] Framework document for QoS Policy Enforcements in Edge Routers for the End-to-End QoS Guaranteed Services: msf2004.126.00
- [7] ETSI Tiphon EMP package: MSF-IA -MEGACO.006-FINAL (msf2003.026)
- [8] Implementation Agreement for MSF Variant Gate Control Protocol: MSF-IA-GCP.001-FINAL (msf2003.114)
- [9] RTP Proxy/FW control protocol: msf2003.113
- [10] COPS usage for RSVP: RFC2749

Appendix A: Example of PIB for Policy Configuration

This section describes examples of PIB for each QoS function in the edge node, such as (a) packet classifier, (b) packet filter, (c) rate meter, (d) packet dropper, (e) queues, (f) packet scheduler, (g) marker and (h) shaper. Note that the explanations of these functions refer to [6].

- Packet classifier and packet filter

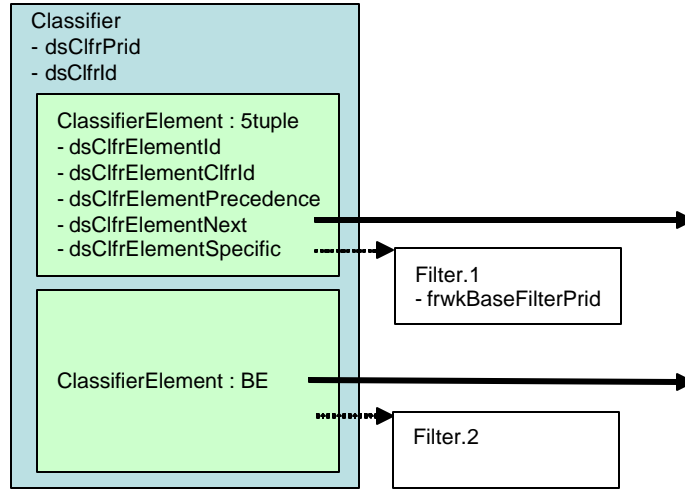


Figure A- 1: Packet classifier and packet filter.

- Rate meter and packet dropper

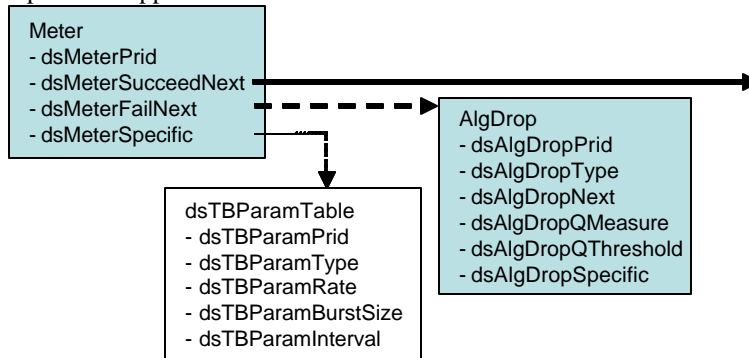


Figure A- 2: Rate meter and packet dropper.

- Marker

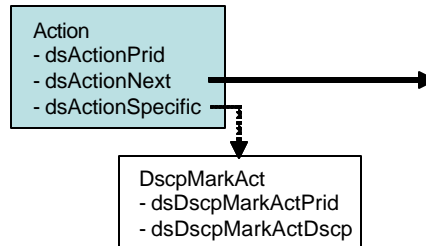


Figure A- 3: Action (marker).

- Queues and scheduler

As an example, the PIB of queue setup (PQ+WFQ: Figure A- 4) is described in the stage format (Figure A-5).

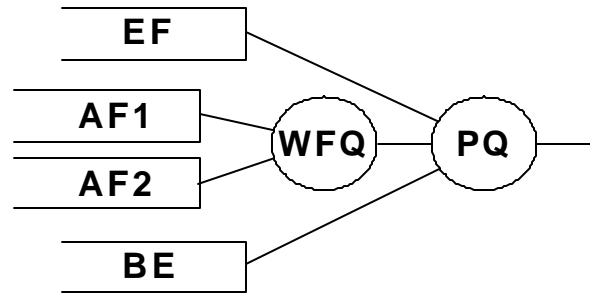


Figure A- 4: Queue.

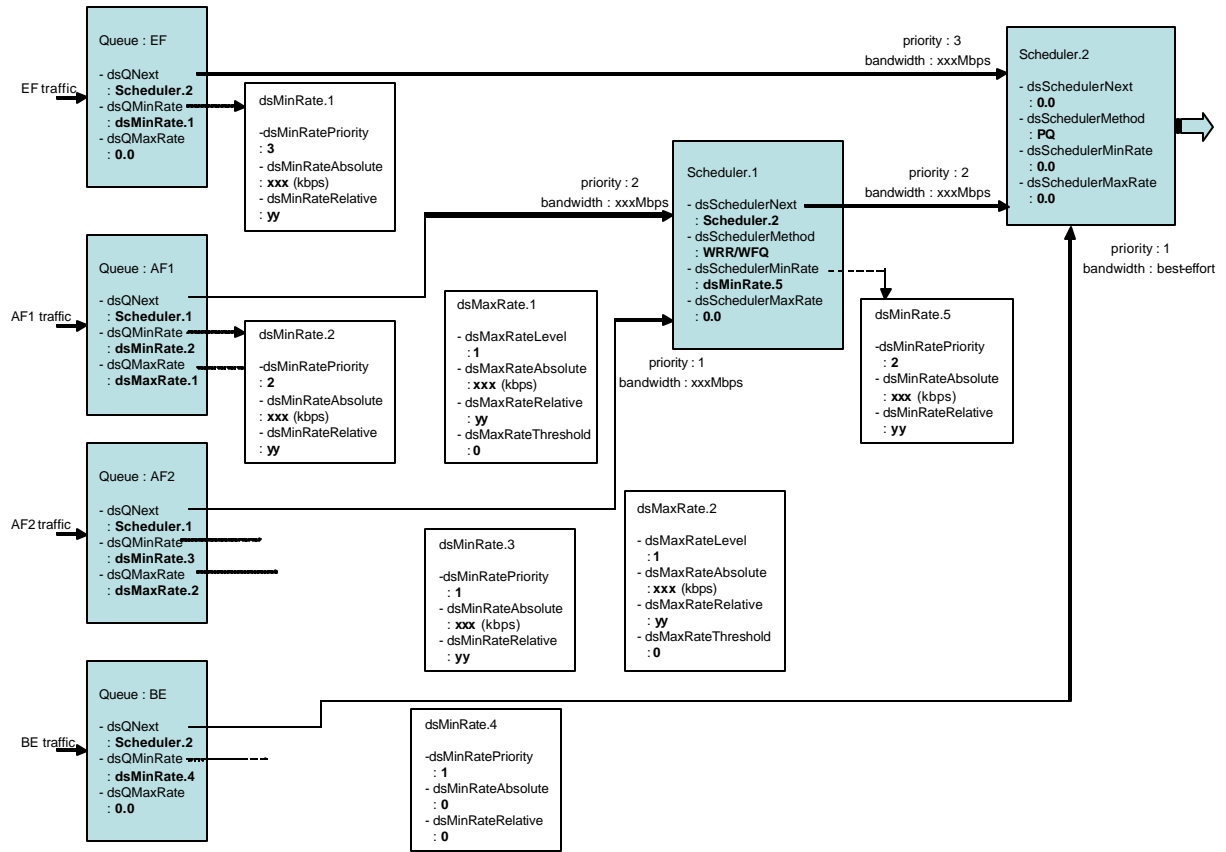


Figure A- 5 : PIB (stage format).

Implementation Agreement for Dynamic Policy Control using COPS-PR

- Example for PIB on DEC message

Table A- 1 shows the DiffServ PIB implementation on DEC message in the dynamic configuration from PDP in Figure F.2.2-1. These PIBs setup packet classifier (source & destination IP address, protocol and source & destination port number), DSCP marking (to AF4 class) and traffic rate limitation (for 1 Mbps) like the PIB elements in Figure A- 6.

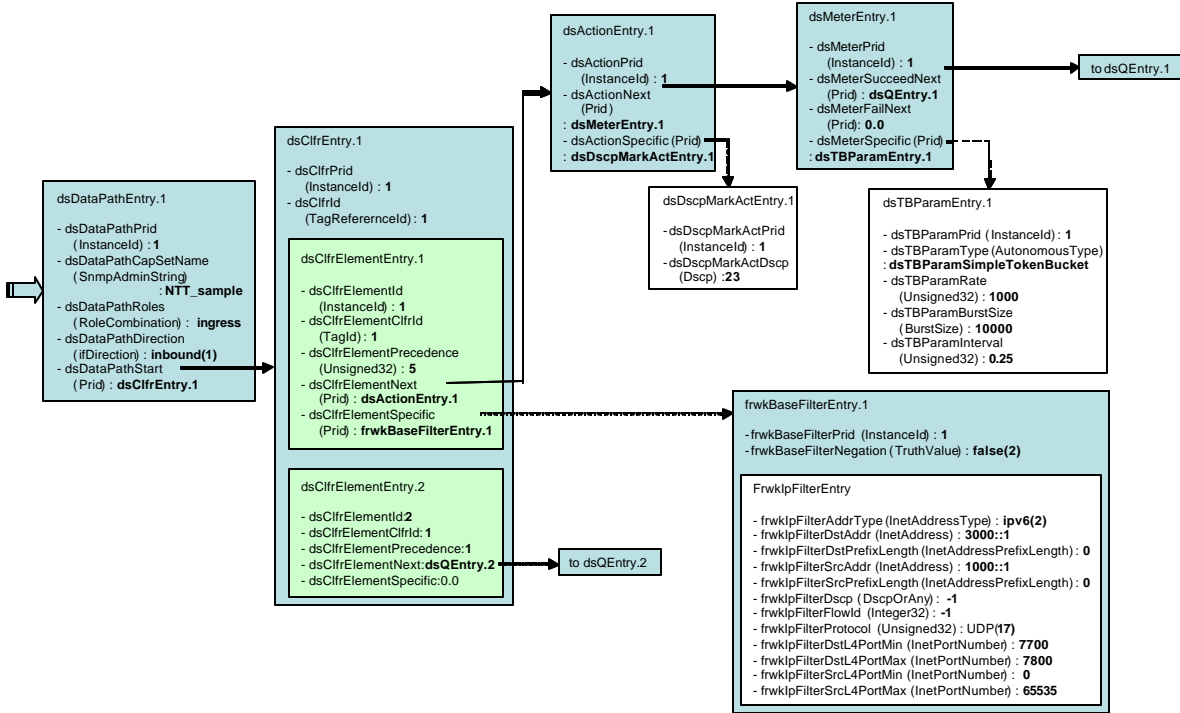


Figure A- 6 : PIB example elements

Table A- 1: DiffServ PIB example on DEC message to install policy for dynamic configuration

+Common Open Policy Service (DEC)			
+Handle Object			
+Contexts: configuration request		(Install)	
+Decision Object: Named Decision Object			
+PRID	1.3.6.1.2.2.2.3.2 {FrwkIpFilterEntry}		
+EPD	frwkIpFilterAddrType	ipv6(2)	
	frwkIpFilterDstAddr	3000::1	
	frwkIpFilterDstPrefixLength	0	
	frwkIpFilterSrcAddr	1000::1	
	frwkIpFilterSrcPrefixLength	0	
	frwkIpFilterDscp	-1	
	frwkIpFilterFlowId	-1	
	frwkIpFilterProtocol	udp(17)	
	frwkIpFilterDstL4PortMin	7700	
	frwkIpFilterDstL4PortMax	7800	
	frwkIpFilterSrcL4PortMin	0	
	frwkIpFilterSrcL4PortMax	65535	
+PRID	1.3.6.1.2.2.2.3.1.1 {frwkBaseFilterEntry.1}		
+EPD	frwkBaseFilterPrid	1	
	frwkBaseFilterNegation	false(2)	
+PRID	1.3.6.1.2.2.4.2.5.1 {dsTBParamEntry.1}		

MSF-IA-COPS-PR.001-FINAL
Implementation Agreement for Dynamic Policy Control using COPS-PR

	+EPD	dsTBParamPrid	1
		dsTBParamType	1.3.6.1.2.2.3.3.1.1 {dsTBParamSimpleTokenBucket}
		dsTBParamRate	1000
		dsTBParamBurstSize	10000
		dsTBParamInterval	0.25
	+PRID	1.3.6.1.2.2.4.2.4.1 {dsMeterEntry.1}	
	+EPD	dsMeterPrid	1
		dsMeterSucceedNext	1.3.6.1.2.2.4.2.11.1 {dsQEntry.1}
		dsMeterFailNext	0.0
		dsMeterSpecific	1.3.6.1.2.2.4.2.5.1 {dsTBParamEntry.1}
	+PRID	1.3.6.1.2.2.4.2.7.1 {dsDscpMarkActEntry.1}	
	+EPD	dsDscpMarkActPrid	1
		dsDscpMarkActDscp	23
	+PRID	1.3.6.1.2.2.4.2.6.1 {dsActionEntry.1}	
	+EPD	dsActionPrid	1
		dsActionNext	1.3.6.1.2.2.4.2.4.1 {dsMeterEntry.1}
		dsActionSpecific	1.3.6.1.2.2.4.2.7.1 {dsDscpMarkActEntry.1}
	+PRID	1.3.6.1.2.2.4.2.3.1 {dsClfrElementEntry.1}	
	+EPD	dsClfrElementId	1
		dsClfrElementClfrId	1
		dsClfrElementPrecedence	5
		dsClfrElementNext	1.3.6.1.2.2.4.2.6.1 {dsActionEntry.1}
		dsClfrElementSpecific	1.3.6.1.2.2.2.3.1.1 {frwkBaseFilterEntry.1}
	+PRID	1.3.6.1.2.2.4.2.3.2 {dsClfrElementEntry.2}	
	+EPD	dsClfrElementId	2
		dsClfrElementClfrId	1
		dsClfrElementPrecedence	1
		dsClfrElementNext	1.3.6.1.2.2.4.2.11.2 {dsQEntry.2}
		dsClfrElementSpecific	0.0
	+PRID	1.3.6.1.2.2.4.2.2.1 {dsClfrEntry.1}	
	+EPD	dsClfrPrid	1
		dsClfrId	1
	+PRID	1.3.6.1.2.2.4.2.1.1 {dsDataPathEntry.1}	
	+EPD	dsDataPathPrid	1
		dsDataPathCapSetName	sample
		dsDataPathRoles	ingress
		dsDataPathDirection	inbound(1)
		dsDataPathStart	1.3.6.1.2.2.4.2.2.1 {dsClfrEntry.1}

Appendix B: Message Contents

This section shows the contents of each message and reflects on previous explanations for objects.

Notes: <>: mandatory objects, []: optional objects, *(): available for several objects

A) Request (REQ) PEP → PDP

```
<REQ> ::= <Common Header>
        <Client Handle>
        <Context = config request>
        *( <ClientSI = Named ClientSI: Request> )
        [ <Integrity> ]
```

```
<ClientSI = Named ClientSI: Request>
    ::= *( <PRID> <EPD> )
```

B) Decision (DEC) PDP → PEP

- Install

```
<DEC> ::= <Common Header>
        <Client Handle>
        *( <Context> )
        <Decision: Flags: Install>
        <Named Decision Data: Provisioning>
        [ <Integrity> ]
```

```
<Named Decision Data> ::= *( <PRID> <EPD> )
```

- Remove

```
<DEC> ::= <Common Header>
        <Client Handle>
        *( <Context> )
        <Decision: Flags: Remove>
        <Named Decision Data: Provisioning>
        [ <Integrity> ]
```

```
<Named Decision Data> ::= *( <PRID> | <PPRID> )
```

- Error

```
<DEC> ::= <Common Header>
        <Client Handle>
        <Error>
        [ <Integrity> ]
```

This message must contain any required “remove” decisions first, followed by any required “install” decisions.

C) Report State (RPT) PEP → PDP

```
<RPT> ::= <Common Header>
        <Client Handle>
        <Report Type>
        *( <ClientSI = Named ClientSI: Report> )
        [ <Integrity> ]
```

<ClientSI = Named ClientSI: Report>
 ::= <[<GPERR>] *(<ErrorPRID> <CPERR> *(<PRID> <EPD>))>

D) Delete Request State (DRQ) PEP → PDP

<DRQ> ::= < Common Header>
 <Client Handle>
 <Reason>
 [<Integrity>]

E) Synchronize State Request (SSQ) PDP → PEP

<SSQ> ::= < Common Header>
 [<Client Handle>]
 [<Integrity>]

F) Client-Open (OPN) PEP → PDP

<OPN> ::= < Common Header>
 <PEPID>
 [<Integrity>]
 [<LastPDPAddr>]

G) Client-Accept (CAT) PDP → PEP

<CAT> ::= < Common Header>
 <KA Timer>
 [<AcctTimer>]
 [<Integrity>]

H) Client-Close (CC) PEP → PDP, PDP → PEP

<CC> ::= < Common Header>
 <Error>
 [<PDPRedirAddr>]
 [<Integrity>]

I) Keep-Alive (KA) PEP → PDP, PDP → PEP

<KA> ::= < Common Header>
 [<Integrity>]

J) Synchronize State Complete (SSC) PEP → PDP

<SSC> ::= < Common Header>
 [<Client Handle>]
 [<Integrity>]

Appendix C. COPS-PR Header

A) Common header

0	1	2	3
Version	Flags	Op-Code	Client-type
Message Length			

Name	Description
Version	Protocol Version = 1
Op-Code	1 = Request (REQ)
	2 = Decision (DEC)
	3 = Report State (RPT)
	4 = Delete Request State (DRQ)
	5 = Synchronize State Request (SSQ)
	6 = Client-Open (OPN)
	7 = Client-Accept (CAT)
	8 = Client-Close (CC)
	9 = Keep-Alive (KA)
	10 = Synchronize Complete (SSC)
C-Num	0x0 Unsolicited Message Flag Bit: unsolicited 0x1 Solicited Message Flag Bit: solicited
Client-type	0 = authentication process in OPN message 2 = DiffServ PIB 0x4000 - 0x7FFF = private use 0x8000 - 0xFFFF = enterprise use managed by IANA
Message Length	Size of message in octets, which includes the standard COPS header and all encapsulated objects.

B) Specific object format

0	1	2	3
Length (Octets)	C-Num	C-Type	
(Object contents)			

Name	Description
Length	Two-octet value
C-Num	1 = Handle
	2 = Context
	3 = In Interface
	4 = Out Interface
	5 = Reason code
	6 = Decision
	7 = LPDP Decision
	8 = Error
	9 = Client Specific Info
	10 = Keep-Alive Timer
	11 = PEP Identification
	12 = Report Type
	13 = PDP Redirect Address

	14 = Last PDP Address
	15 = Accounting Timer
	16 = Message Integrity
C-Type	See section 6.
(Object contents)	See section 6.

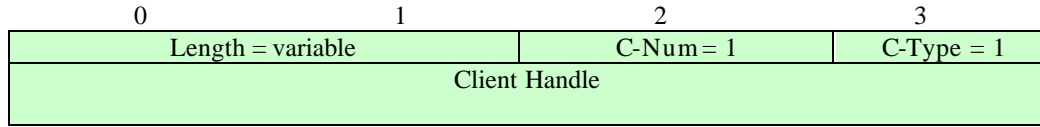
C) Specific sub-object format

0	1	2
Length (Octets)	S-Num	S-Type
(Sub-Object contents)		

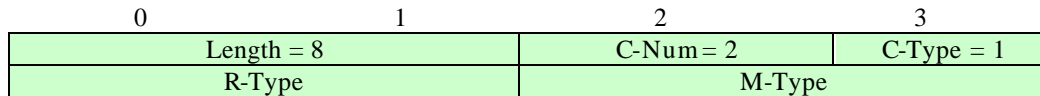
Name	Description
Length	Two-octet value
S-Num	1 = Complete PRID, Length = variable.
	2 = Prefix PRID, Length = variable.
	3 = EPD, Length = variable.
	4 = GPERR, Length = 8.
	5 = CPERR, Length = 8.
	6 = ErrorPRID, Length = variable.
S-Type	1 = BER (for BER)
(Sub-Object contents)	See section 7.

Appendix D. COPS-PR Object Formats

A) Handle object



B) Context object (context)



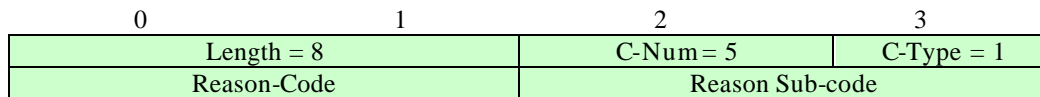
R-Type (Request type flag)

Bits	Description
0x01	Incoming-message/admission control request
0x02	Resource-allocation request
0x04	Outgoing-message request
0x08	Configuration request

M-Type (Message type)

Client-specific 16-bit values of protocol message types

C) Reason object (reason)



Reason Code:

Code	Description
1	Unspecified
2	Management
3	Preempted (Another request state takes precedence)
4	Tear (Used to communicate a signaled state removal)
5	Timeout (Local state has timed-out)
6	Route change (Change invalidates request state)
7	Insufficient resources (No local resource available)
8	BM's directive (BM decision caused the delete)
9	Unsupported decision (BM decision not supported)
10	Synchronize handle unknown
11	Transient handle (stateless event)
12	Malformed decision (could not recover)
13	Unknown COPS Object from BM: Sub-code (octet 2) contains unknown object's C-Num and (octet 3) contains unknown object's C-Type.

D) Decision object (decision)

1. Decision Flags (mandatory)

0	1	2	3
Length = variable	C-Num = 6	C-Type = 1	
Command-Code		Flags	

Name	Description
Command-code	0 = NULL Decision (No configuration data available) 1 = Install (Admit request/Install configuration) 2 = Remove (Remove request/Remove configuration)
Flags	0x01 = Trigger error (Trigger error message if set) 0x02 = Request-state flag

2. Named Decision Data

0	1	2	3
Length = variable	C-Num = 6	C-Type = 5	
<Sub-Object>			

Sub-object (see section 7):

- Install (when “Decision flags” indicates “install”).
This object is included in DEC messages in case of policies installing. The PEP conveys policies to install using the combination of “PRID” and “EPD” within this object.
 - S-Num = 1 (Complete PRID), S-Type = 1 (BER), Length = variable.
 - S-Num = 3 (EPD), S-Type = 1 (BER), Length = variable.

- Remove (when “Decision flags” indicates “remove”).
This object is included in DEC messages in case of policies removing. The PEP conveys policies to remove using “Prefix PRID”.
 - S-Num = 2 (Prefix PRID), S-Type = 1 (BER), Length = variable.

Named Decision Data object conveys policy information bases (PIBs). (see section 9)

E) Error object (Error)

0	1	2	3
Length = variable	C-Num = 8	C-Type = 1	
Error-Code		Error Sub-code	

Error-Code:

Code	Cause
1	Bad handle
2	Invalid handle reference
3	Bad message format (Malformed Message)
4	Unable to process (server gives up on query)
5	Mandatory client-specific info missing
6	Unsupported client-type
7	Mandatory COPS object missing
8	Client Failure
9	Communication Failure

10	Unspecified
11	Shutting down
12	Redirect to Preferred Server
13	Unknown COPS Object: Sub-code (octet 2) contains unknown object's C-Num and (octet 3) contains unknown object's C-Type.
14	Authentication Failure
15	Authentication Required

F) Client-specific Information object (ClientSI)

1. Named ClientSI

0	1	2	3
Length = variable	C-Num = 9	C-Type = 2	
<Sub-Object>			

Sub-object (see section 7):

- Within request message
 - This object is included in REQ messages. The clients specify policies provisioning using the combination of “PRID” and “EPD” within this object.
 - S-Num = 1 (Prefix PRID), S-Type = 1 (BER), Length = variable.
 - S-Num = 3 (EPD), S-Type = 1 (BER), Length = variable.

- Within report message
 - This object is included in RRT messages. The clients report the cause of troubles using the following COPS-PR objects within this object.
 - S-Num = 2 (Complete PRID), S-Type = 1 (BER), Length = variable.
 - S-Num = 3 (EPD), S-Type = 1 (BER), Length = variable.
 - S-Num = 4 (GPERR), S-Type = 1 (for BER), Length = 8.
 - S-Num = 5 (CPERR), S-Type = 1 (for BER), Length = 8.
 - S-Num = 6 (ErrorPRID), S-Type = 1 (BER), Length = variable.

Named ClientSI object conveys policy information bases (PIBs). (see section9)

G) Keep-Alive Timer object (KATimer)

0	1	2	3
Length = 8	C-Num = 10	C-Type = 1	
0x0000	KA Timer Value		

Keep-Alive timer value;

- 1 to 65535: the number of seconds
- 0: set to infinity.

H) PEP Identification object (PEPID)

0	1	2	3
Length = variable	C-Num = 11	C-Type = 1	
ASCII strings			

The PEP Identifier object is used to identify the edge node in the remote BM.

I) Report-Type object (Report-Type)

0	1	2	3
Length = 8		C-Num = 12	C-Type = 1
Report-Type		0x0000	

Report-Type:

Code	Description
1	Success : Decision was successful at the edge node
2	Failure : Decision could not be completed by edge node
3	Accounting: Accounting update for an installed state

J) PDP Redirect Address (PDPRedireAddr)

- Mandatory for IPv4 address

0	1	2	3
Length = 12		C-Num = 13	C-Type = 1
Ipv4 Address format			
0x00		TCP Port Number	

- Optional for IPv6 address

0	1	2	3
Length = 12		C-Num = 13	C-Type = 2
Ipv6 Address format			
0x00		TCP Port Number	

K) PDP Last PDP Address object (LastPDPAddr)

- Mandatory for IPv4 address

0	1	2	3
Length = 12		C-Num = 14	C-Type = 1
Ipv4 Address format			
0x00		TCP Port Number	

- Optional for IPv6 address

0	1	2	3
Length = 12		C-Num = 14	C-Type = 2
Ipv6 Address format			
0x00		TCP Port Number	

L) Accounting Timer object (AcctTimer)

0	1	2	3
Length = 8		C-Num = 15	C-Type = 1
0x00		ACCT Timer Value	

M) Message Integrity object (Integrity)

0	1	2	3
Length = variable		C-Num = 16	C-Type = 1
Key ID			
Sequence Number			
Keyed Message Digest			
.....			

Appendix E. COPS-PR Policy Provisioning object Formats

A) Complete Provisioning Instance Identifier (PRID)

0	1	2	3
Length = variable		S-Num = 1 (PRID)	S-Type = 1 (BER)
Instance Identifier			

B) Prefix PRID (PPRID)

0	1	2	3
Length = variable		S-Num = 2 (Prefix PRID)	S-Type = 1 (BER)
Prefix PRID			

C) Encoded Provisioning Instance Data (EPD)

0	1	2	3
Length = variable		S-Num = 3 (EPD)	S-Type = 1 (BER)
BER Encoded PRI Value			

D) Global Provisioning Error object (GPERR)

0	1	2	3
Length = 8		S-Num = 4 (GPERR)	S-Type = 1 (BER)
Error-Code		Error Sub-code	

Error-Code:

Code	Description
availMemLow(1)	
availMemExhausted(2)	
unknownASN.1Tag(3)	The erroneous tag type SHOULD be specified in the Error Sub-Code field.
maxMsgSizeExceeded(4)	COPS message (transaction) was too big.
unknownError(5)	
maxRequestStatesOpen(6)	No more Request-States can be created by the PEP (in response to a DEC message attempting to open a new Request-State).
invalidASN.1Length(7)	The ASN.1 object length was incorrect.
invalidObjectPad(8)	Object was not properly padded.
unknownPIBData(9)	Some of the data supplied by the PDP is unknown/unsupported by the PEP (but otherwise formatted correctly). PRC-specific error codes are to be used to provide more information.
unknownCOPSPRObject(10)	Sub-code (octet 2) contains unknown object's S-Num and (octet 3) contains unknown object's S-Type.
malformedDecision(11)	Decision could not be parsed.

E) PRC Class Provisioning Error object (CPERR)

0	1	2	3
Length = 8		S-Num = 5 (CPERR)	S-Type = 1 (BER)
Error-Code		Error Sub-code	

Error-Code:

Code	Description
priSpaceExhausted(1)	No more instances may currently be installed in the given class.
priInstanceInvalid(2)	The specified class instance is currently invalid prohibiting
attrValueInvalid(3)	The specified value for identified attribute is illegal. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.
attrValueSupLimited(4)	The specified value for the identified attribute is legal but not currently supported by the device. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.
attrEnumSupLimited(5)	The specified enumeration for the identified attribute is legal but not currently supported by the device. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.
attrMaxLengthExceeded(6)	The overall length of the specified value for the identified attribute exceeds device limitations. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.
attrReferenceUnknown(7)	The class instance specified by the policy instance identifier does not exist. The sub-code SHOULD identify the OID sub-identifier of the attribute associated with the error.
priNotifyOnly(8)	The class is currently only supported for use by request or report messages prohibiting decision installation.
unknownPrc(9)	Attempt to install a PRI of a class not supported by PEP.
tooFewAttrs(10)	Received PRI has fewer attributes than required.
invalidAttrType(11)	Received PRI has an attribute of the wrong type.
deletedInRef(12)	Deleted PRI is still referenced by other (non) deleted PRIs.
priSpecificError(13)	The Error Sub-code field contains the PRC specific error code.

F) Error PRID object (ErrorPRID)

0	1	2	3
Length = variable		S-Num =6 (ErrorPRID)	S-Type = 1 (BER)
Instance Identifier			

Appendix F. Call Flows

F.1 End-to-end call flows

These call flows target COPS-PR behavior between BM and Edge node triggered by the CA.

F.2 Call flows for normal operation

F.2.1 Restart (set up a static configuration)

F.2.1.1 No previous state exists

Behavior in the establishment of a COPS-PR session:

- The edge node issues OPN to BM with a PEPID, which indicates its identifier.

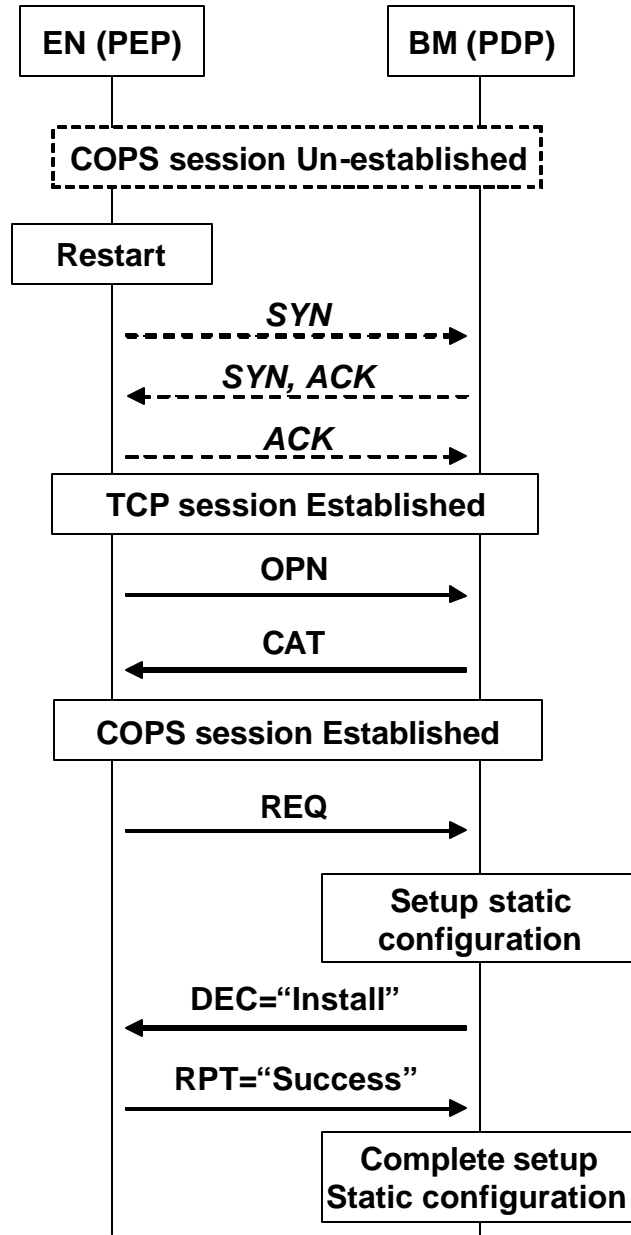


Figure F.2.1-1: Static configuration setup procedure.

F.2.1.2 Previous state exists

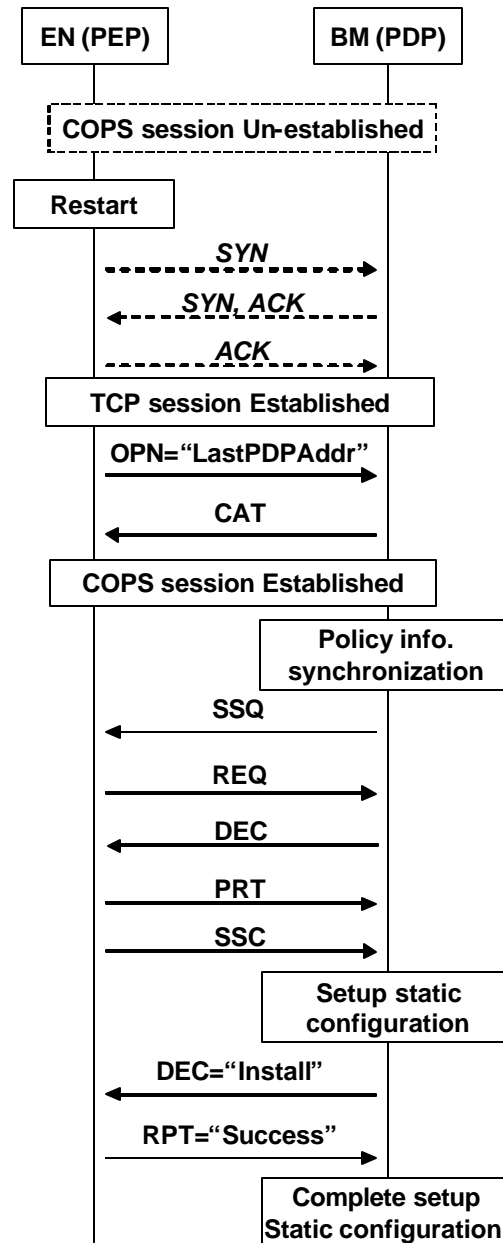


Figure F.2.1-2: Static configuration setup procedure.

Send trigger of SSQ;

- After the edge node reboots or COPS-PR session is established and sends an OPN message with LastPDPAddr object, the BM synchronizes policy information between the BM and the edge node by sending SSQ to the edge node. After BM issues CAT, the BM issues SSQ and collects policy information in the edge node at the point.
- The edge node that receives SSQ issues REQ to the BM with the capability or a policy that has already been set up.
- After the edge node sends all policy information, it issues SSC to the BM. This is the notification that the process has been completed.

F.2.2 Set up a dynamic configuration

Edge node control from BM:

- The BM controls policies in the edge node by sending DEC and installs or deletes policies using PIB.
- The edge node issues RPT to the BM to report policy control results.

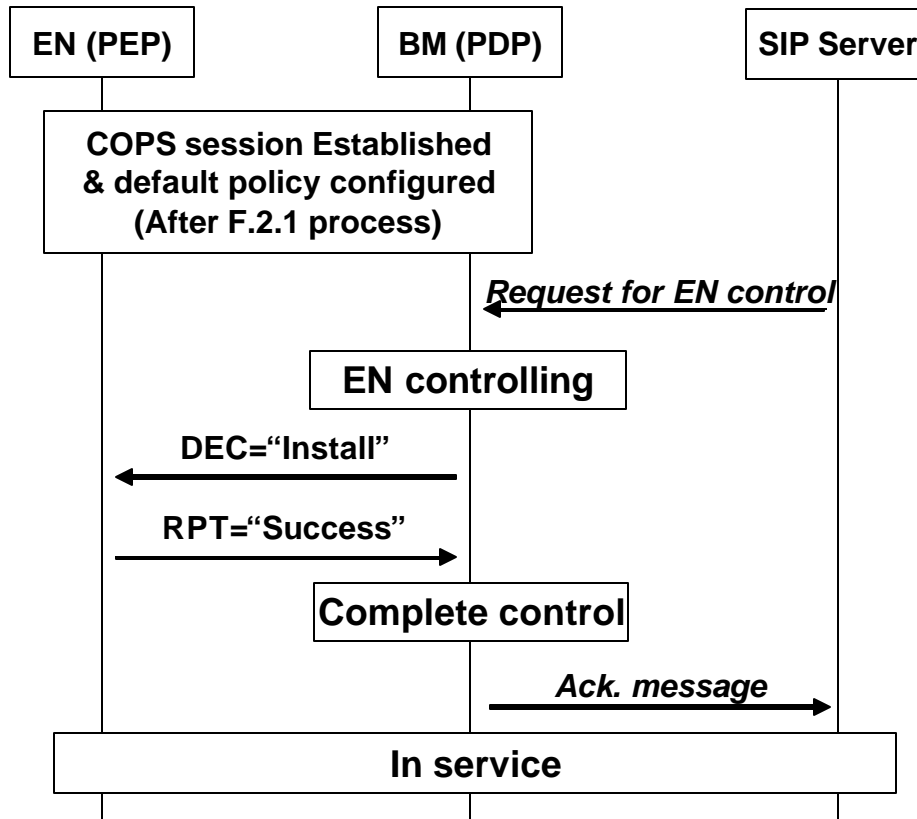


Figure F.2.2-1: setup dynamic configuration

F.2.3 Keep Alive call flows

Keep Alive:

- The edge node issues KA to the BM for every 1/4 of the “Maximum Keep Alive Interval” time.
- The BM replies to the KA as soon as it receives it (within the receive message time-out).
- The Client-Type of the common header of KA is set to NULL.

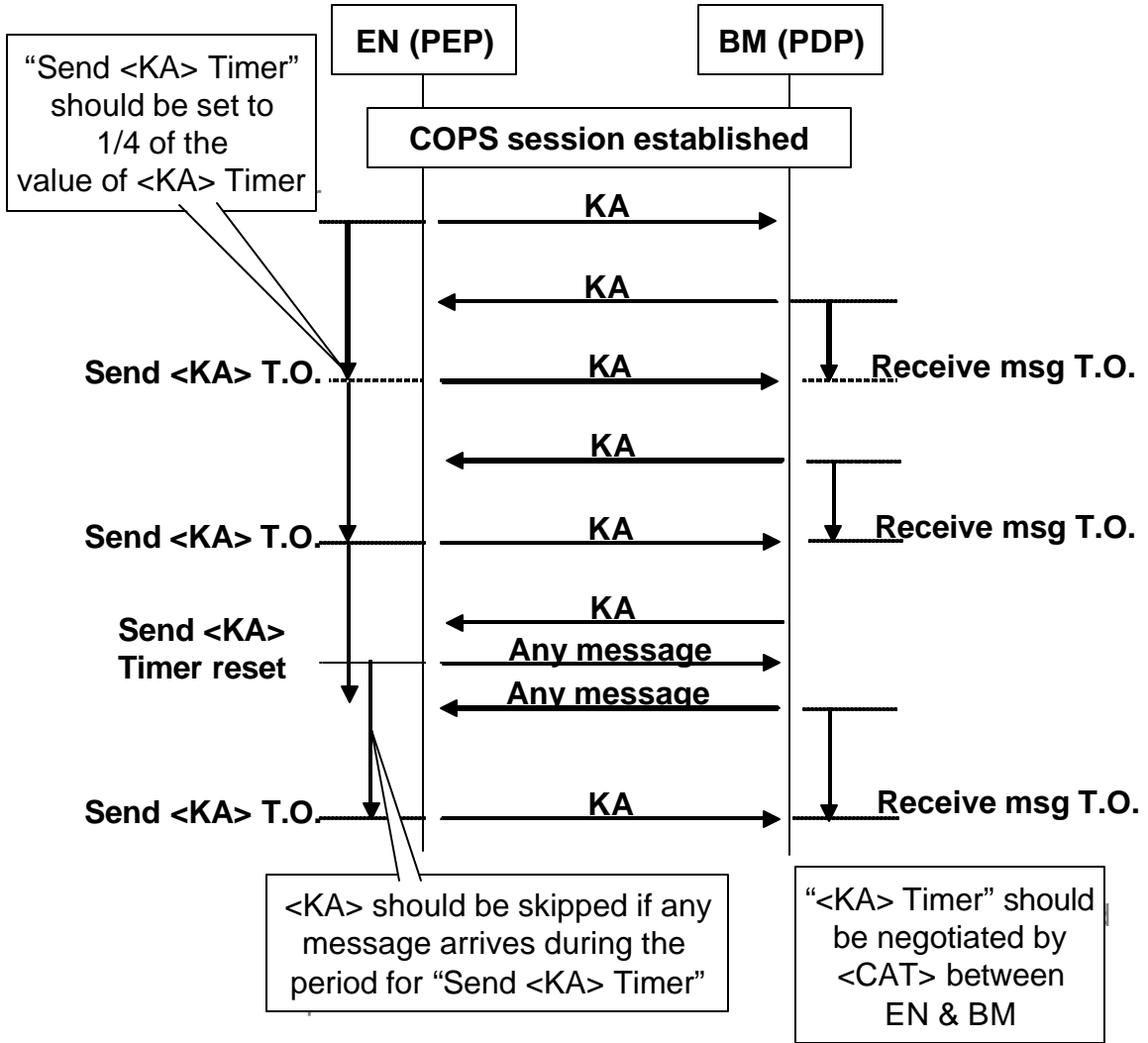


Figure F.2.3-1: Keep Alive call flows

F.2.4 COPS-PR session close triggered by edge nodes

Disconnect COPS-PR session:

- An entity that intends to disconnect the COPS-PR session issues CC to the peering entity. CC has information about the cause of this disconnection.

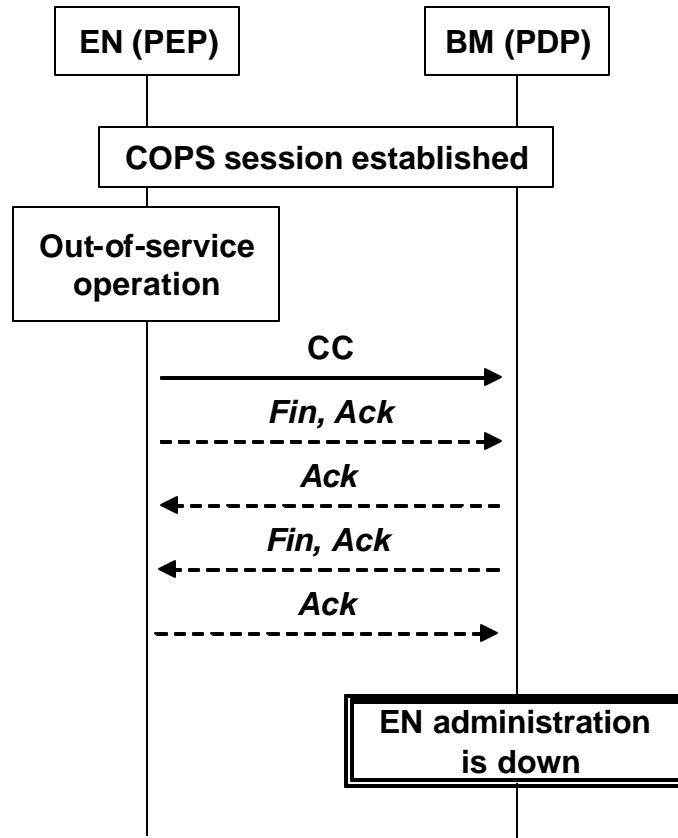


Figure F.2.4-1: COPS-PR session close triggered by Edge nodes

F.3 Call flows for error operation

F.3.1 TCP session establishment failure

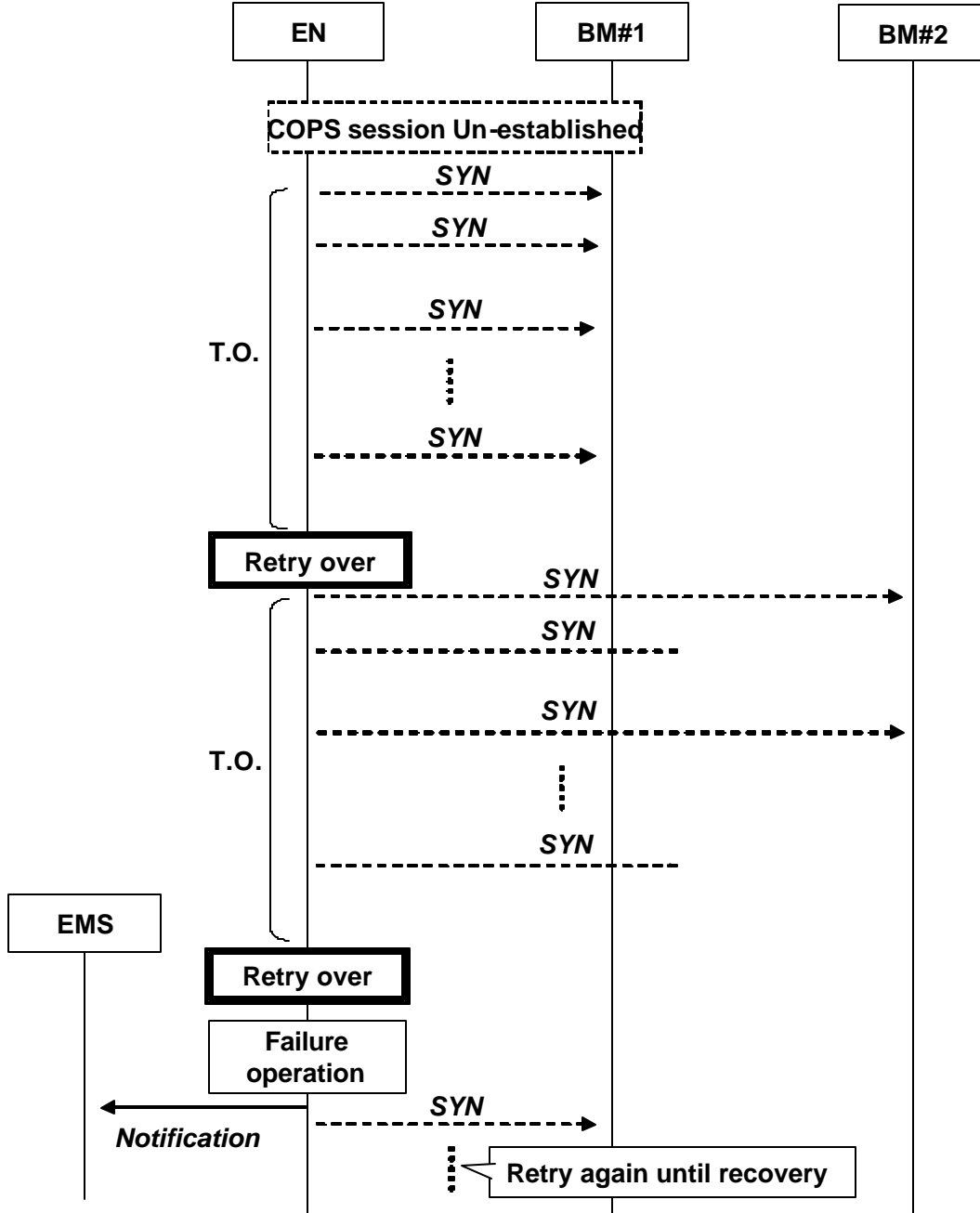


Figure F.3.1-1: TCP session establishment failure.

“Failure operation”: the behavior of the edge node in the case of a failure to decide to give up the retry process and send notification to the element management system.

F.3.2 <OPN>/<CAT> Time out

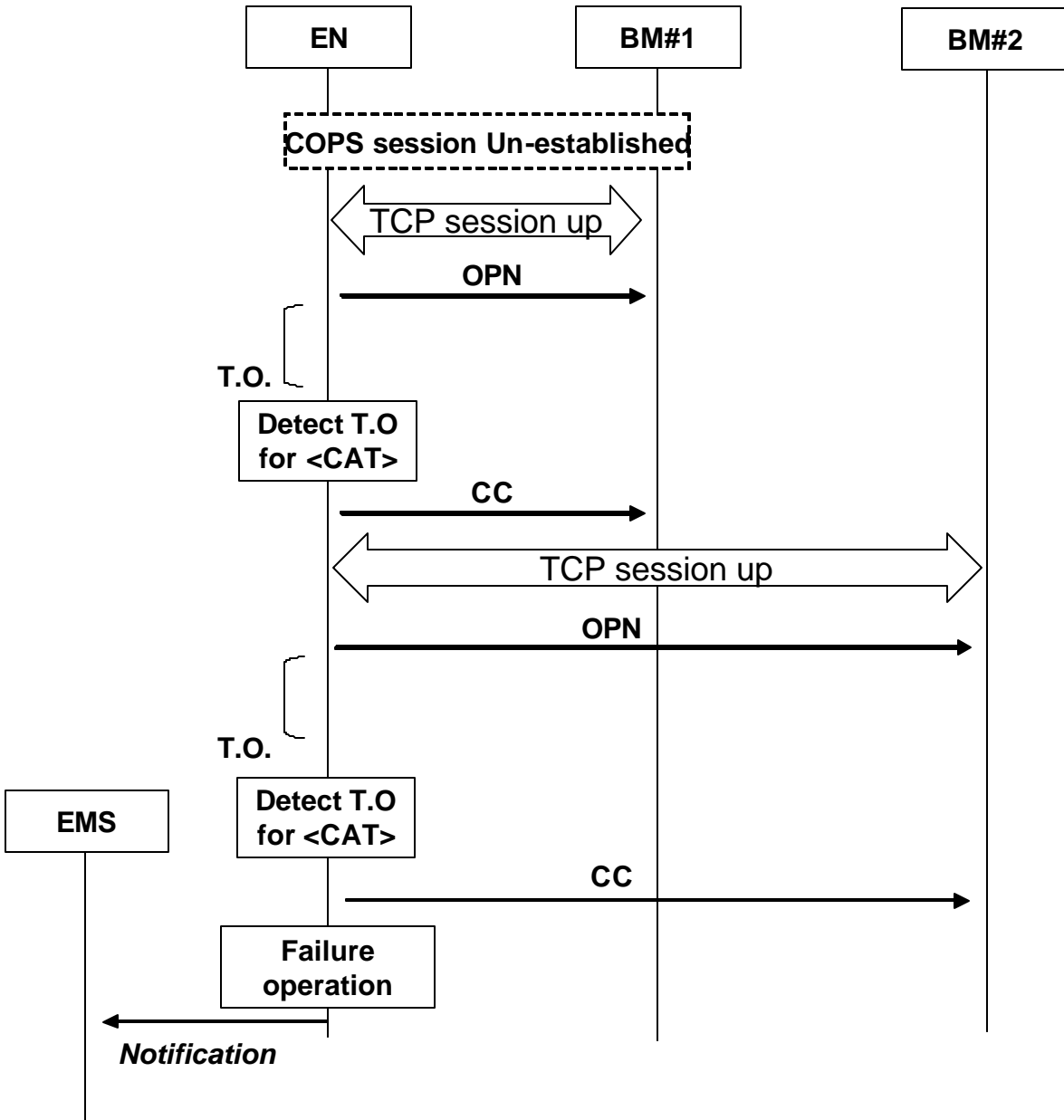


Figure F.3.2-1: <OPN>/<CAT> failure.

“Failure operation”: the behavior of the edge node in the case of a failure to decide to give up the retry process and send notification to the element management system.

F.3.3 <OPN> error

(1) Administrative error

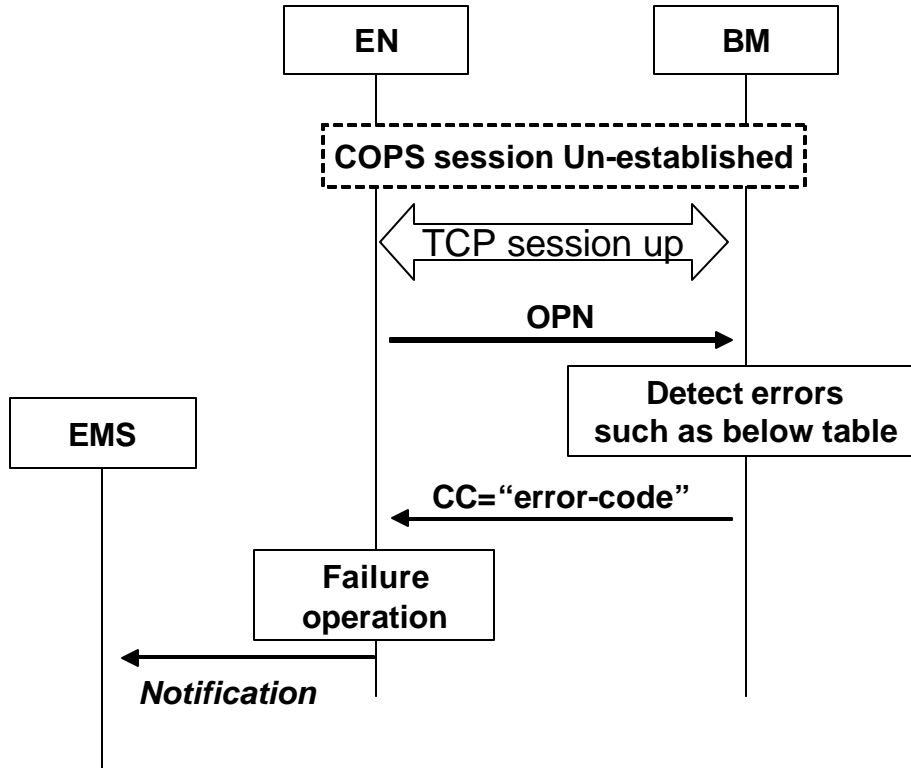


Figure F.3.3-1: <OPN> error (administrative error).

Table F.3.3-1 <CC> error-code table (administrative error).

Code	Cause
1	Bad handle
2	Invalid handle reference
3	Bad message format (malformed message)
5	Mandatory client-specific info. missing
6	Unsupported client-type
7	Mandatory COPS object missing
8	Client failure
10	Unspecified
13	Unknown COPS object: sub-code (octet 2) contains an unknown object's C-Num and (octet 3) contains an unknown object's C-Type.

“Failure operation”: the behavior of the edge node in the case of a failure to decide to give up the retry process and send notification to the element management system.

(2) Retry

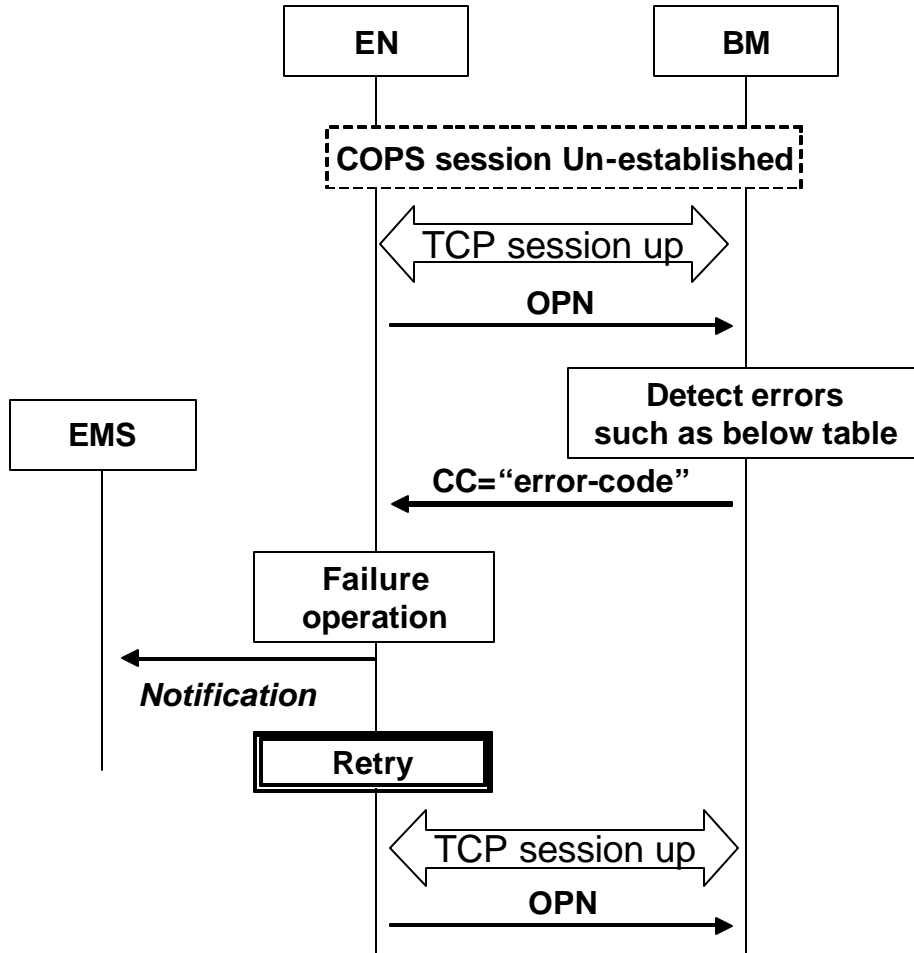


Figure F.3.3-2: <OPN> error (retry).

Table F.3.3-2 <CC> error-code table (retry)

Code	Cause
4	Unable to process (server gives up on query)
9	Communication failure
11	Shutting down
14	Authentication failure
15	Authentication required

“Failure operation”: the behavior of the edge node in the case of a failure to send notification to the element management system.

(3) Connection to redundant BM

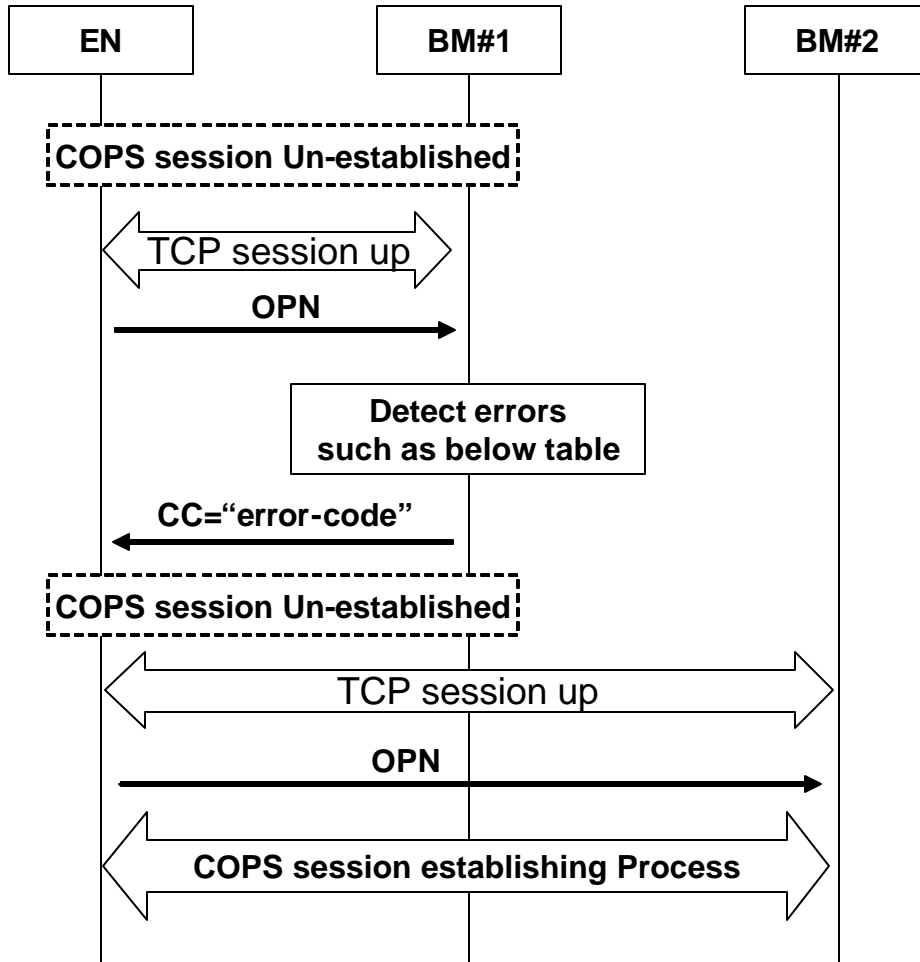


Figure F.3.3-3: <OPN> error (connection to redundant BM).

Table F.3.3-3 <CC> Error-Code table (connection to alternative BM).

Code	Cause
12	Redirect to preferred server

F.3.4 <REQ> time out

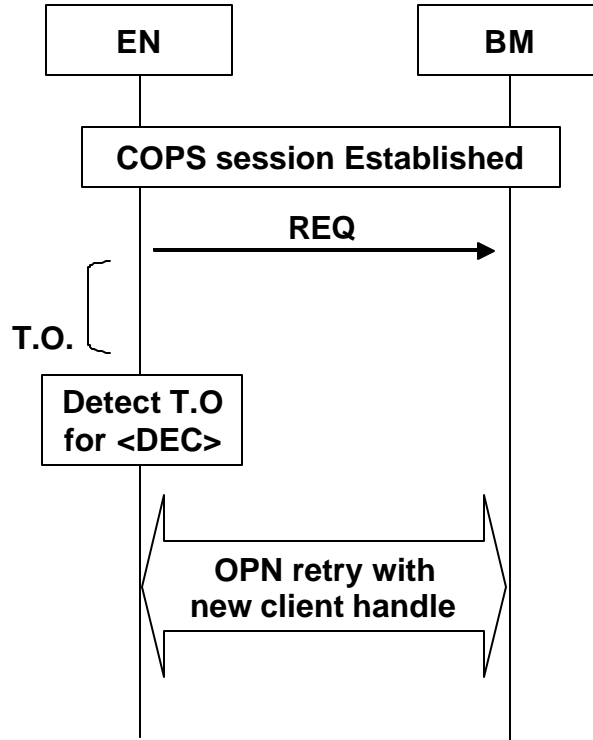


Figure F.3.4-1: <REQ> time out.

F.3.5 <REQ> rejection

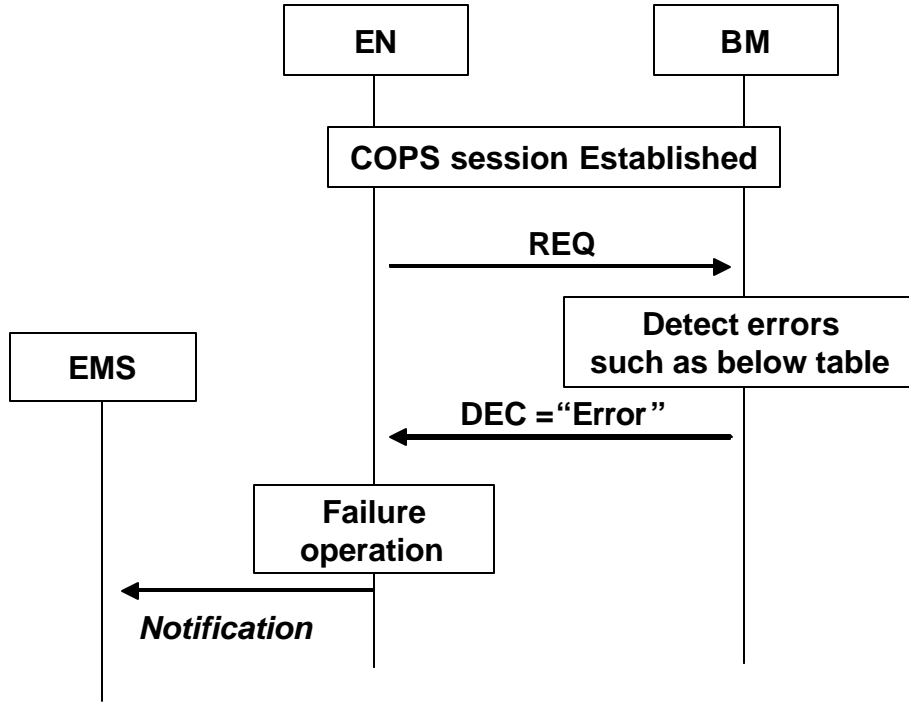


Figure F.3.5-1: <REQ> rejection.

Table F.3.5-1 <DEC>error -code table.

Code	Cause	Status
1	Bad handle	
2	Invalid handle reference	
3	Bad message format (malformed message)	
4	Unable to process (server gives up on query)	
5	Mandatory client-specific info missing	
6	Unsupported client-type	
7	Mandatory COPS object missing	
8	Client failure	
9	Communication failure	
10	Unspecified	
11	Shutting down	
13	Unknown COPS object: sub-code (octet 2) contains an unknown object's C-Num and (octet 3) contains an unknown object's C-Type.	
14	Authentication failure	
15	Authentication required	

“Failure operation”: the behavior of the edge node in the case of a failure to decide to give up the retry process and send notification to the element management system.

F.3.6 <DEC> time out

<DEC> is send triggered by <REQ> or a request from a SIP server.

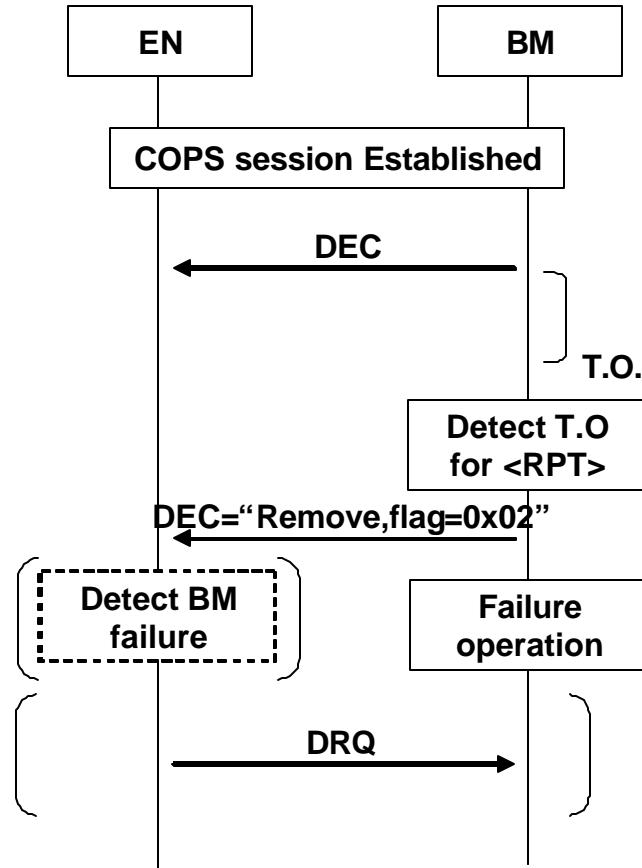


Figure F.3.6-1: <DEC> time out.

F.3.7 Policy setup error

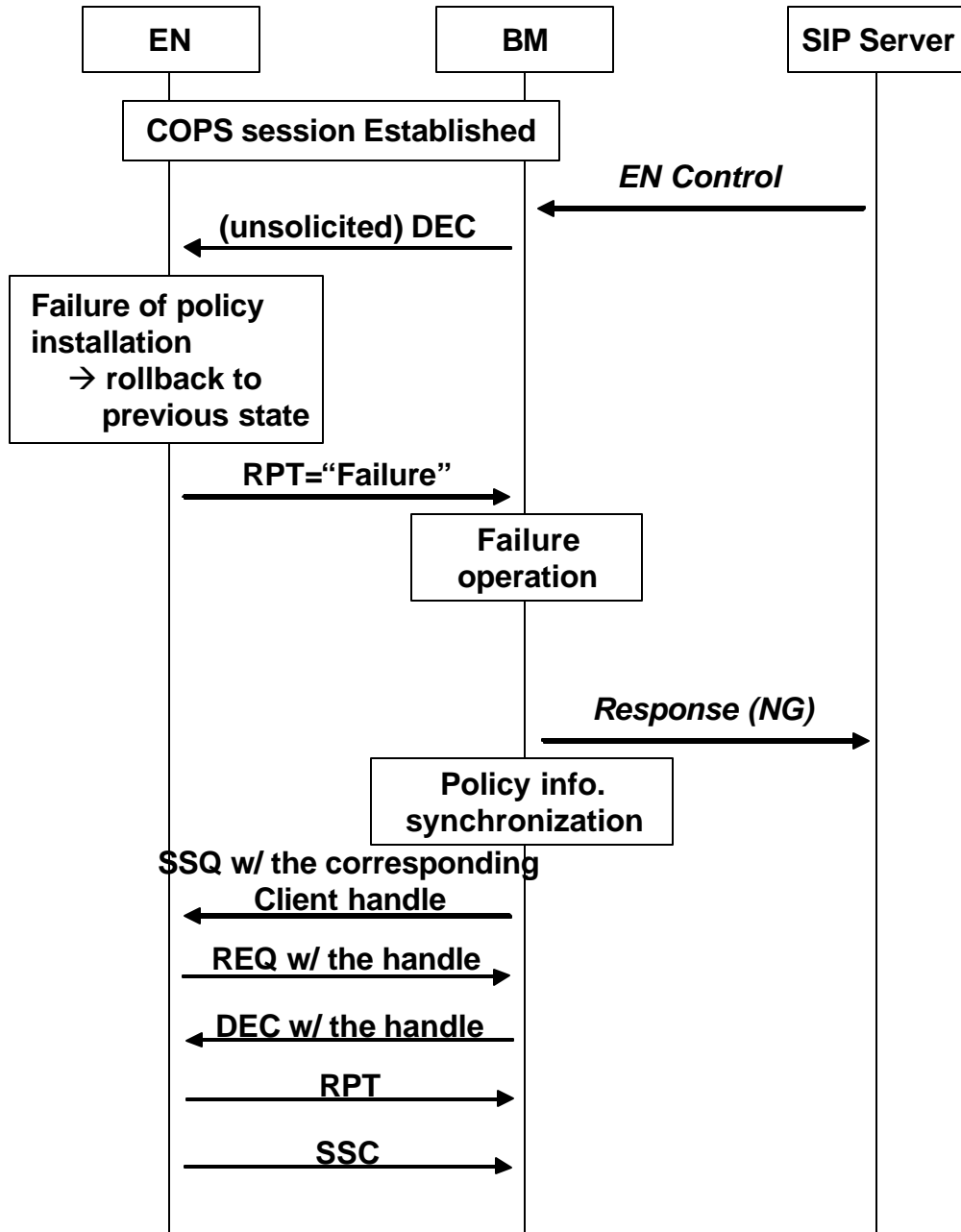


Figure F.3.7-1: Policy setup error.

F.3.8 Keep Alive time-out by edge node

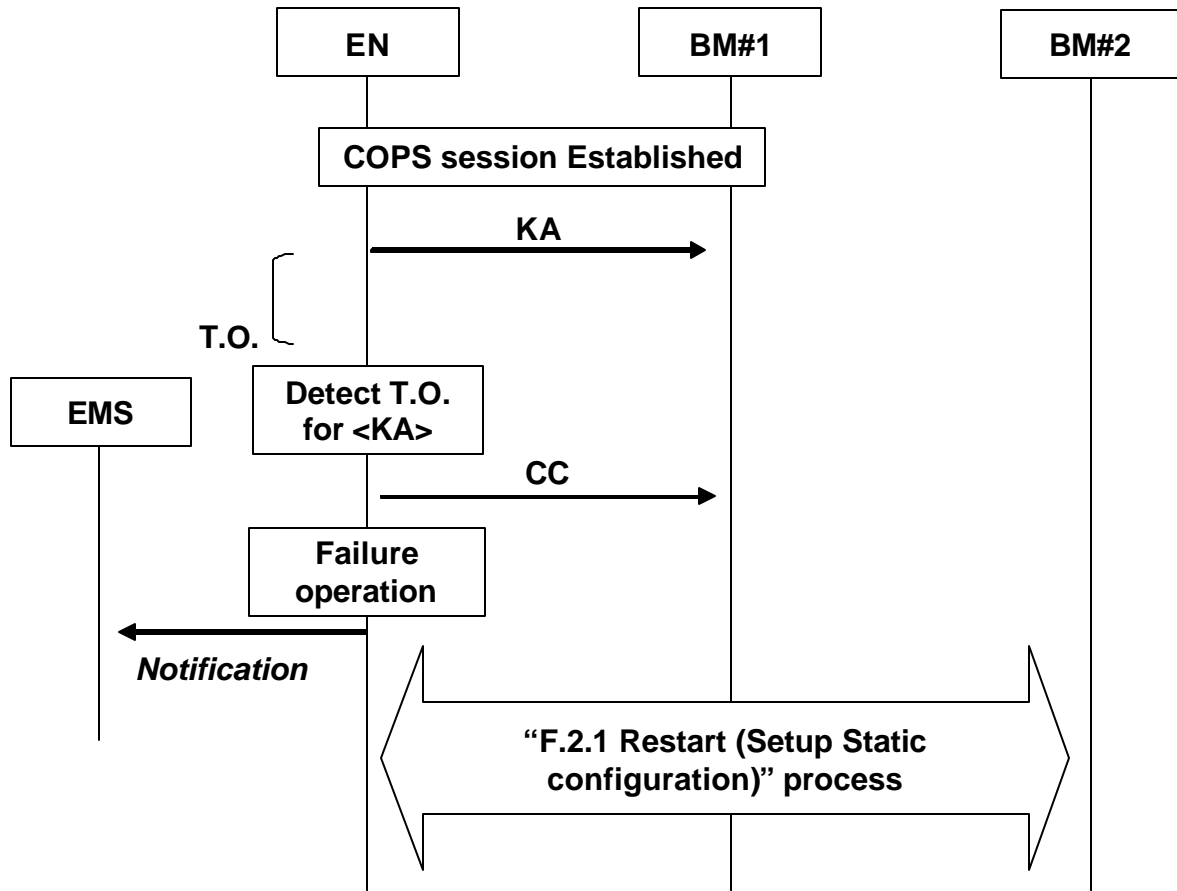


Figure F.3.8-1: Keep Alive time-out by edge node

"Failure operation": the behavior of the edge node in the case of a failure to decide to give up the retry process and send notification to the element management system.

F.3.9 Keep Alive Time Out by BM

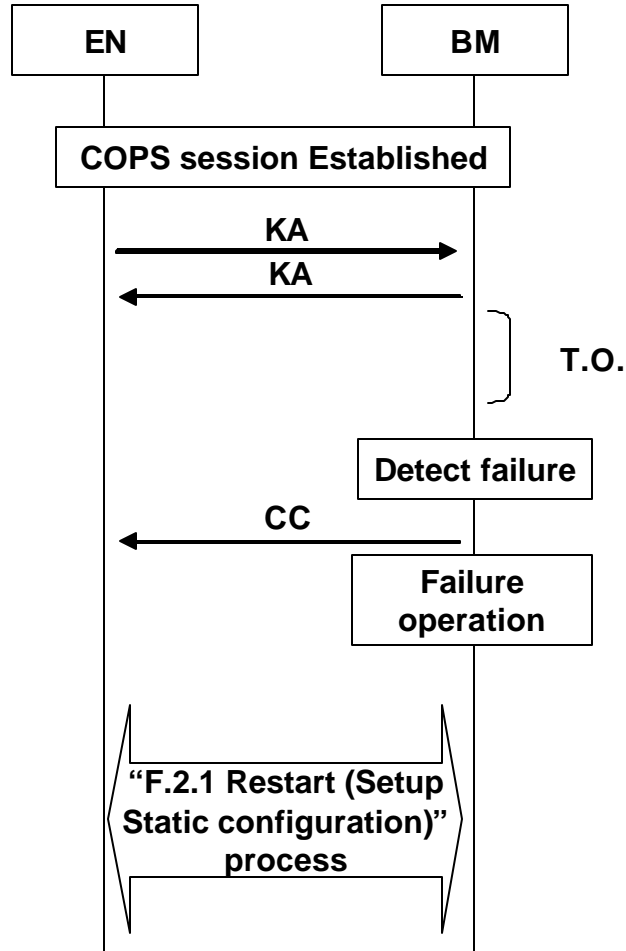


Figure F.3.9-1: Keep Alive time out by BM.

F.4 Call flows fault operation

F.4.1 Forced COPS-PR session closure by BM

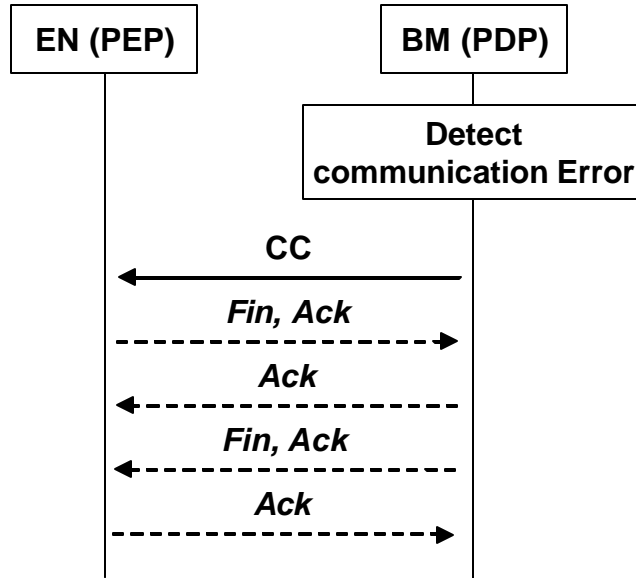


Figure F.4.1-1: Forced COPS-PR session closure by BM.

F.4.2 Call flow during edge node failure

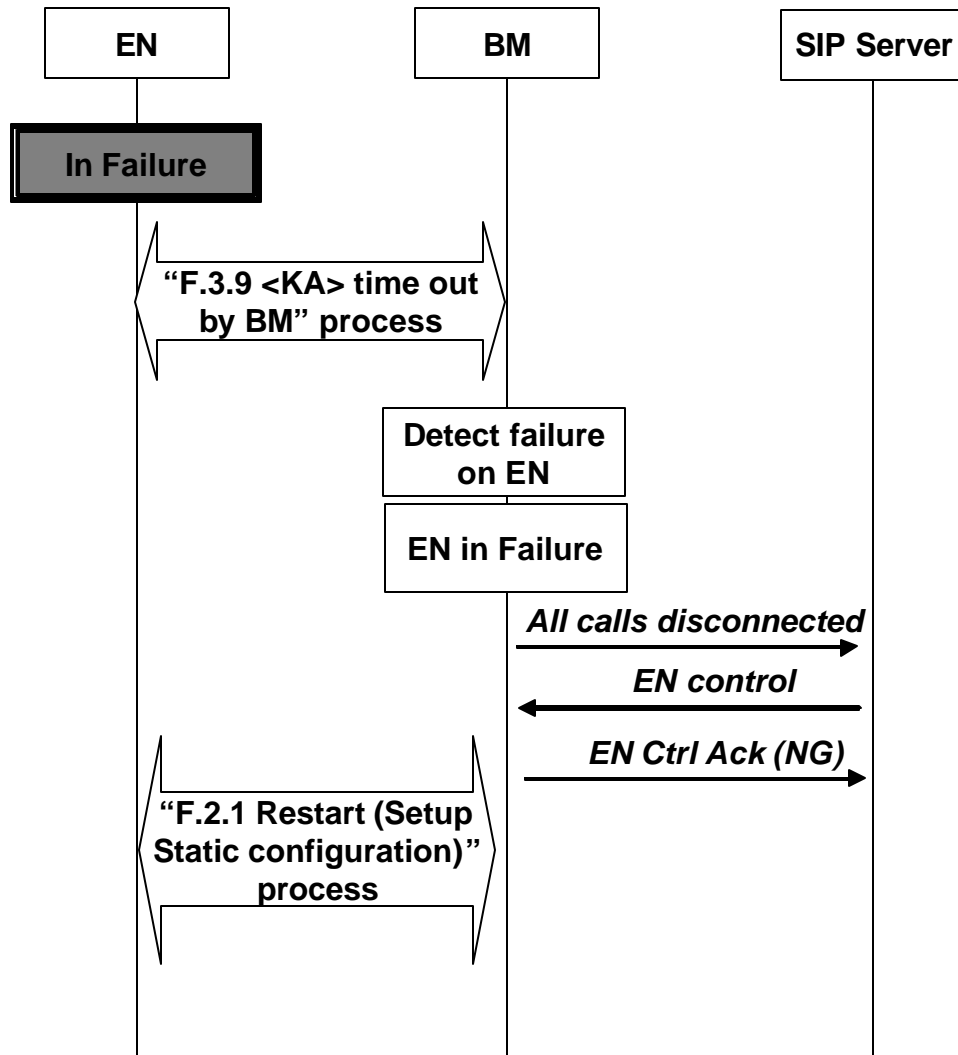


Figure F.4.2-1: Call flow during edge node failure.

F.4.3 BM failure

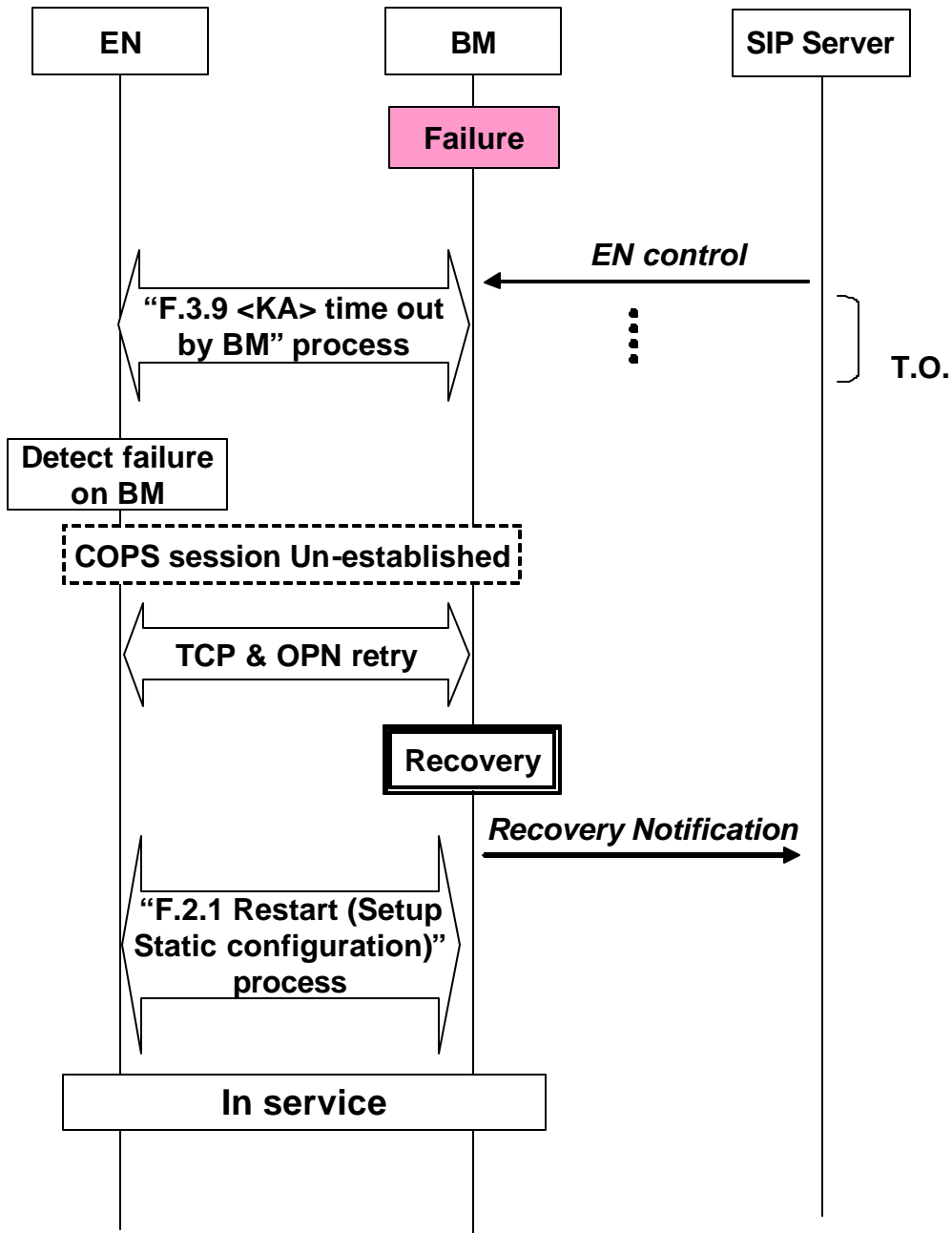


Figure F.4.3-1: BM failure.

Appendix G. Authentication Operation

(For further study)