INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

T.122

(02/98)

SERIES T: TERMINALS FOR TELEMATIC SERVICES

Multipoint communication service – Service definition

ITU-T Recommendation T.122

(Previously CCITT Recommendation)

ITU-T T-SERIES RECOMMENDATIONS

TERMINALS FOR TELEMATIC SERVICES

For further details, please refer to ITU-T List of Recommendations.

ITU-T RECOMMENDATION T.122

MULTIPOINT COMMUNICATION SERVICE – SERVICE DEFINITION

Summary

This Recommendation defines a multipoint data delivery service for use in the audiographics and audiovisual conferencing service. It provides the mechanism for multipoint aware applications to send data to all or a subset of the group with a single send primitive and to force, if desired, a uniformly sequenced reception of data at all users. It also provides a token mechanism to allow applications to control scarce resources or do multi-application signalling and synchronization. These services are provided in a manner that is independent of the underlying network connections.

The following changes have been incorporated into this Recommendation:

– Support for unreliable data transfer was added.

– Under certain conditions, MCS protocol version is allowed to differ between providers within a domain.

– The MCS-DOMAIN-PARAMETERS service primitive was added.

– Parameters were added to MCS-SEND-DATA and MCS-UNIFORM-SEND-DATA service primitives to facilitate data reassembly by the user application.

– Minor changes have been made to the Recommendation format.

Source

ITU-T Recommendation T.122 was revised by ITU-T Study Group 16 (1997-2000) and was approved under the WTSC Resolution No. 1 procedure on the 6th of February 1998.

FOREWORD

ITU (International Telecommunication Union) is the United Nations Specialized Agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of the ITU. The ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Conference (WTSC), which meets every four years, establishes the topics for study by the ITU-T Study Groups which, in their turn, produce Recommendations on these topics.

The approval of Recommendations by the Members of the ITU-T is covered by the procedure laid down in WTSC Resolution No. 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

The ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. The ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, the ITU had not received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

© ITU 1998

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the ITU.

CONTENTS

		Page
1	Scope	1
2	References	1
3	Definitions	2
4	Abbreviations	3
5	Conventions	3
5.1	Primitive parameters conventions	3
5.2	Primitive notations	3
6	Model of MCS	3
7	MCS connections and building the domain	4
7.1	Establishing connections and binding them to the domain	4
7.2	Attaching application users to a domain	6
7.3	MCS-DOMAIN-PARAMETERS	6
8	MCS channels	7
8.1	Multicast channels	7
8.2	Single-member channels	7
8.3	Private channels	8
8.4	Channel Id numbering	8
9	MCS data transfer	8
9.1	Simple Send	8
9.2	Uniformly sequenced data transfer	9
9.3	Send data with responses (For further study)	9
9.4	Unreliable data transfer	10
10	Token management	10
10.1	Exclusive event control and transfer	10
10.2	Event coordination	11
10.3	Token Id Numbering	11
11	Introduction to MCS service primitives	11
11.1	MCS domain management primitives	11
11.2	MCS Channel Management primitives	11
11.3	MCS data transfer primitives	11
11.4	MCS token management primitives	11

Page

12 MCS domain management primitives 12
12.1 MCS-CONNECT-PROVIDER 12
12.1.1 Function 12
12.1.2 Types of primitives and their parameters 12
12.1.3 Sequence of primitives 13
12.2 MCS-DISCONNECT-PROVIDER 14
12.2.1 Function 14
12.2.2 Types of primitives and their parameters 14
12.2.3 Sequence of primitives 14
12.3 MCS-ATTACH-USER 15
12.3.1 Function 15
12.3.2 Types of primitives and their parameters 15
12.3.3 Sequence of primitives 15
12.4 MCS-DETACH-USER 15
12.4.1 Function 15
12.4.2 Types of primitives and their parameters 16
12.4.3 Sequence of primitives 16
12.5 MCS-DOMAIN-PARAMETERS 16
12.5.1 Function 16
12.5.2 Types of primitives and their parameters 17
12.5.3 Sequence of primitives 17

13 MCS channel management primitives 18
13.1 MCS-CHANNEL-JOIN 18
13.1.1 Function 18
13.1.2 Types of primitives and their parameters 18
13.1.3 Sequence of primitives 18
13.2 MCS-CHANNEL-LEAVE 18
13.2.1 Function 18
13.2.2 Types of primitives and their parameters 19
13.2.3 Sequence of primitives 19
13.3 MCS-CHANNEL-CONVENE 19
13.3.1 Function 19
13.3.2 Types of primitives and their parameters 20
13.3.3 Sequence of primitives 20
13.4 MCS-CHANNEL-DISBAND 20
13.4.1 Function 20
13.4.2 Types of primitives and their parameters 20

Page

13.4.3 Sequence of primitives 21
13.5 MCS-CHANNEL-ADMIT 21
13.5.1 Function 21
13.5.2 Types of primitives and their parameters 21
13.5.3 Sequence of primitives 22
13.6 MCS-CHANNEL-EXPEL 22
13.6.1 Function 22
13.6.2 Types of primitives and their parameters 22
13.6.3 Sequence of primitives 22
14 MCS data transfer primitives 23
14.1 MCS-SEND-DATA 23
14.1.1 Function 23
14.1.2 Types of primitives and their parameters 23
14.1.3 Sequence of primitives 24
14.2 MCS-UNIFORMLY-SEQUENCED-SEND-DATA 24
14.2.1 Function 24
14.2.2 Types of primitives and their parameters 24
14.2.3 Sequence of primitives 25
15 MCS token management primitives 25
15.1 MCS-TOKEN-GRAB 25
15.1.1 Function 25
15.1.2 Types of primitives and their parameters 26
15.1.3 Sequence of primitives 26
15.2 MCS-TOKEN-INHIBIT 26
15.2.1 Function 26
15.2.2 Types of primitives and their parameters 26
15.2.3 Sequence of primitives 27
15.3 MCS-TOKEN-GIVE 27
15.3.1 Function 27
15.3.2 Types of primitives and their parameters 27
15.3.3 Sequence of primitives 28
15.4 MCS-TOKEN-PLEASE 28
15.4.1 Function 28
15.4.2 Types of primitives and their parameters 28
15.4.3 Sequence of primitives 29
15.5 MCS-TOKEN-RELEASE 29
15.5.1 Function 29

Page

15.5.2 Types of primitives and their parameters 29
15.5.3 Sequence of primitives 29
15.6 MCS-TOKEN-TEST 30
15.6.1 Function 30
15.6.2 Types of primitives and their parameters 30
15.6.3 Sequence of primitives 30
Annex A – Domain establishment, data transfer and release phases examples 30
A.1 MCS domain establishment phase 30
A.2 MCS data transfer phase 35
A.3 MCS connection release phase 37
Appendix I – Distributed token control 38

Recommendation T.122

MULTIPOINT COMMUNICATION SERVICE – SERVICE DEFINITION

(revised in 1998)

1 Scope

The Multipoint Communication Service (MCS) is a generic service designed to support highly interactive multimedia conferencing applications. It supports full-duplex multipoint communication among an arbitrary number of connected application entities over a variety of networks as specified in Recommendation T.123. This version of MCS utilizes only the basic mode of Recommendation T.123. MCS provides efficient multicast message sequencing and token management features by means of an MCS provider, to which users are connected either directly or through other MCS providers.

MCS offers the following features:

a) Flexible modes of data transfer:
   - Broadcast, with flow control;
   - Request/Response.

b) Multipoint addressing:
   - One to all;
   - One to sub-group;
   - One to one.

c) Multipoint routing of data:
   - Shortest path to each receiver;
   - Uniform sequencing of data, where all users receive the same data in the same sequence.

d) Tokens are provided for resource contention resolution.

e) Network independent:
   - For fully reliable data transmission, MCS assumes the use of error-free transport connections that provide flow control (see Recommendation T.123).

MCS provides information to conference participants about other participants only in cases where this information cannot be provided by the participants themselves. For example, members of a domain are not informed when a new member joins (the new member can do so if it wishes), whereas they are informed by MCS when a member detaches, since the departing member may have no opportunity to inform the others.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

– ITU-T Recommendation F.710 (1991), *General principles for audiographic conference service*.

– ITU-T Recommendation T.123 (1996), *Network specific data protocol stacks for multimedia conferencing*.

3 Definitions

This Recommendation defines the following terms:

3.1 MCS provider: An entity (i.e. active element) of an MCS subsystem, interacting directly with application entities above and transport entities below. The services it provides and the services it uses are modelled in terms of abstract primitives. An MCS provider communicates with peer MCS providers.

3.2 MCS domain: A tree of MCS connections among MCS providers or, in the degenerate case, part of a single MCS provider. The MCS providers involved are said to *host* the domain. A domain sets the boundary for data transfer among attached MCS users.

3.3 MCS domain selector: An octet string that distinguishes multiple domains hosted by the same MCS provider. The creation of MCS providers, their configuration to host one or more domains, and the internal structure of domain selectors are all local matters.

3.4 top MCS provider: In every domain a single MCS provider will become the Top MCS Provider and will be the exclusive manager of all the domain's channel, user Ids and token resources.

3.5 MCS connection: A specified set of transport connections, managed as one unit, between a pair of MCS providers. Each end of an MCS Connection is bound to a respective MCS domain selector. One end of each MCS Connection is designated to be hierarchically superior to the other.

3.6 MCS user: An application entity obtaining services from an MCS provider. Once they have been attached to the same domain, MCS users can transfer data in point-to-point or multicast fashion.

3.7 MCS Service Access Point (MCSAP): A point through which an MCS user accesses an MCS provider.

3.8 MCS attachment: A multi-connection end point within an MCSAP. Each MCS attachment is bound to an MCS domain selector. An MCS user may attach through an MCSAP to domains hosted by the MCS provider.

3.9 MCS user Id: A short identifier, unique within an MCS domain, that distinguishes between MCS attachments. A user with multiple attachments to the same domain, through the same or different SAPs, has equally many user Ids.

3.10 control-MCSAP: A unique MCSAP per MCS provider (distinguished by some local means) to which the MCS Connect and MCS Disconnect primitives are restricted. The MCS user of a Control-MCSAP does not have direct access to data transfer through the MCS Connection end points it contains; this is limited to the MCS provider. It is a local option whether a Control-MCSAP may also contain MCS attachments like ordinary MCSAPs and allow the MCS user to invoke other MCS primitives.

3.11 MCS channel: A channel is a domain-wide address. Channels are used for MCS data unit distribution lists. All users who are members of the same channel will receive data sent to that channel.

3.12 MCS private channel: A channel with an authorized group of users. Only they may send and receive data over the channel. Attempts by unauthorized users to invoke an MCS primitive involving a private channel as parameter will fail.

3.13 MCS private channel manager: An MCS user who convenes a private channel and regulates its authorized user group. If the manager detaches, the private channel is disbanded.

3.14 version 3 or V3 summit: The nodes in a domain hierarchy which use MCS protocol version 3 or greater and are either:

- a Top Provider; or
- have an upward connection path to a version 3 Top Provider that flows only through other version 3 nodes.

4 Abbreviations

This Recommendation uses the following abbreviations:

MCS Multipoint Communication Service

MCSAP MCS Service Access Point

MCSPDU MCS Protocol Data Unit

MCU Multipoint Control Unit (see Recommendation F.710)

SAP Service Access Point

5 Conventions

5.1 Primitive parameters conventions

The primitive parameters defined in this Recommendation will use the following key:

M Parameter is mandatory

C Parameter is conditional

U Parameter is a user option

Blank Parameter is absent

(=) Value of the parameter is identical to the value of the corresponding parameter of the preceding primitive.

5.2 Primitive notations

The primitive services referred to in the text will be in italics.

6 Model of MCS

MCS establishes a multipoint domain over point-to-point MCS connections. Within that multipoint domain, an application client can send data to different members of the domain and have access to tokens for resource contention resolution.

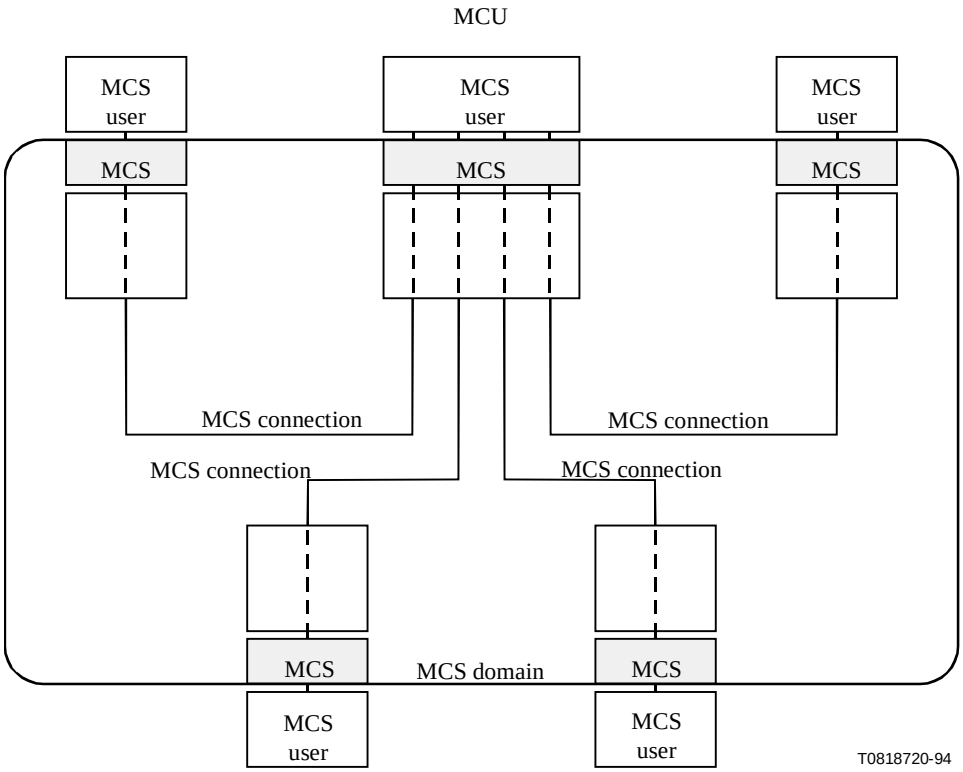


Figure 1/T.122 – MCS model

The MCS user first establishes an MCS connection between its MCS provider and a remote MCS provider. This connection is bound to a domain to which users can attach. Users at other sites can set up MCS connections to sites already part of the domain and bind them to the same domain.

NOTE – If all communication is to be among applications at one site, connections are not necessary, since only one MCS provider is involved.

Once the domain is established, the MCS user then joins the appropriate channels it requires for receiving data. The use of these channels is application dependent. Tokens are provided for managing the resources available to the client.

7 MCS connections and building the domain

For multipoint communication between remote sites to occur, their MCS providers must first be connected and bound to the same domain. A multipoint domain is a hierarchical structure composed of MCS connections that connect MCS providers together. A domain is constructed by binding connections to it.

7.1 Establishing connections and binding them to the domain

A Controller sets up an MCS connection through its local MCS provider to a remote site using the *connect provider* service. An MCS connection is a set of transport connections managed as one unit between two MCS providers. The application client communicates with its MCS agent through a Multipoint Communication Service Access Point (MCSAP).

An MCS connection is responsible for delivering MCS data units between MCS providers. An MCS provider can have multiple MCS connections.

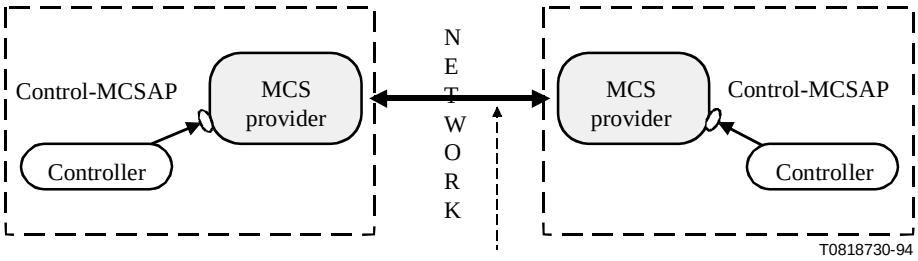
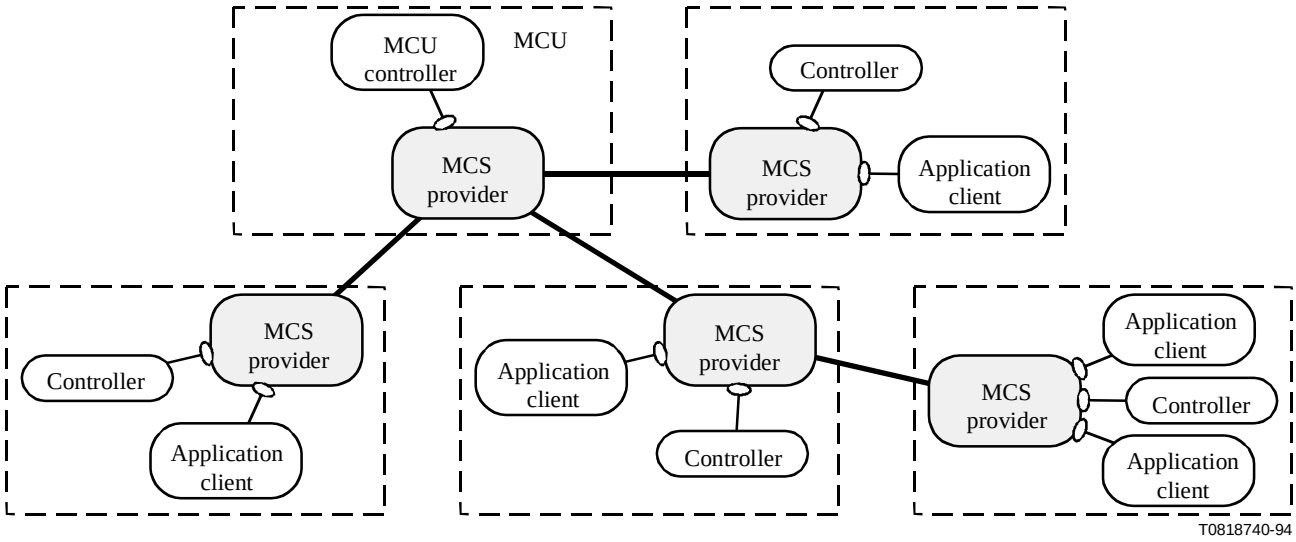


Figure 2/T.122 – MCS connection phase

A controller expecting to receive *connect provider* indications should request a Control-MCSAP. MCS will then know how to direct such indications when they arrive.



NOTE – Some MCS providers have multiple MCS connections and some have multiple attached users. Some MCS providers reside on MCUs, while others reside on terminals.

Figure 3/T.122 – Five-site network with multiple MCS connections

The optional user data field of the *connect provider* messages can be used to exchange security information, for example passwords or any other data required by the multipoint application. The *connect provider* request is initiated at one site and approved or rejected by another site. *Connect provider* indications are locally directed to the Control-SAP of the destination provider.

A multipoint domain is organized in a hierarchical structure. The MCS provider at the top of the hierarchy is the Top MCS provider. It is during the domain setup stage that the network hierarchy is defined. Within the *connect provider* request is a field defining whether this connection to be attached is going up to a higher level, or down to a lower one.

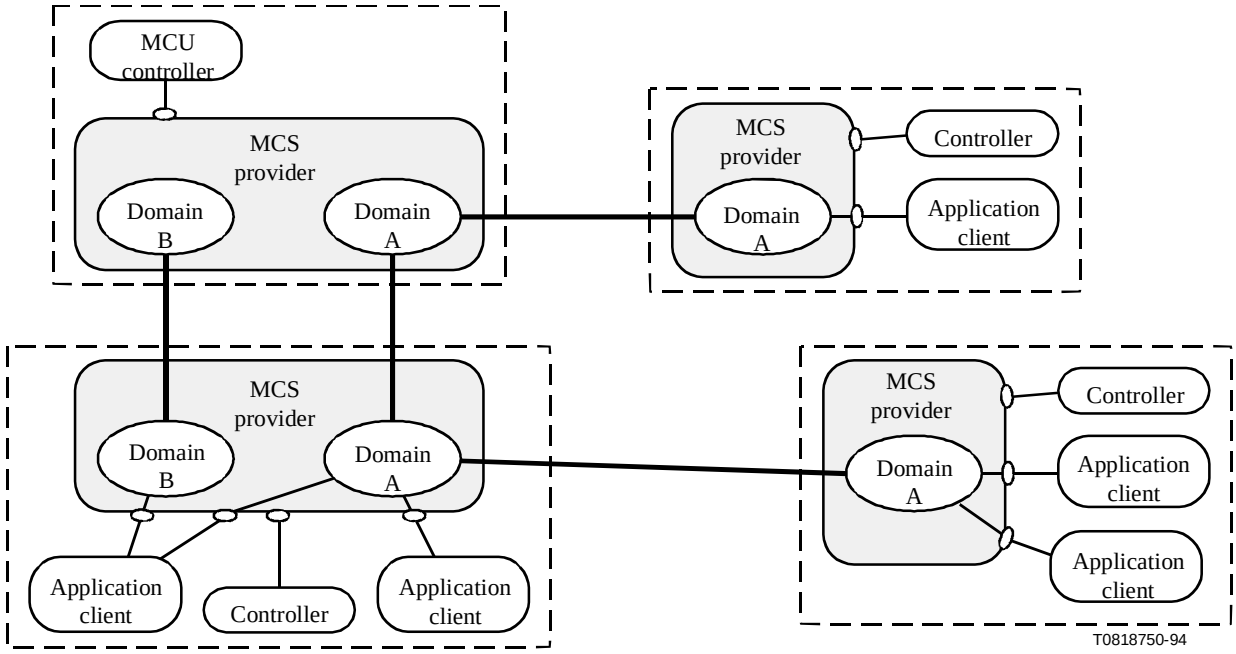
NOTE – The domain hierarchical structure is necessary for the allocation of user Ids, for the use of tokens, and for uniformly sequenced data.

The *disconnect provider* service is used to leave a domain and to disconnect transport connections.

7.2 Attaching application users to a domain

Application clients become attached to a domain using the *attach user* service. It differs from connecting providers in that it does not require the up/down field for defining domain hierarchy. A SAP is attached to no more than one provider, and is used by no more than one application client at any given time.

An application client automatically receives from MCS in the *attach-user* confirm its own unique user identifier for the domain duration. There is some structural limit to the number of application clients in a domain or channel.



NOTE – Application clients can be attached through multiple SAPs to more than one domain.

Figure 4/T.122 – Multiple domains

The *detach user* primitive is used by an application client to leave a domain.

7.3 MCS-DOMAIN-PARAMETERS

With the exception of protocol version, domain parameters are identical throughout a domain. They are negotiated in the first *connect provider* primitive which sets up the domain. MCS providers joining a domain after the initial two have no say in determining domain parameters. An application client can determine domain parameters by examining the response of a *connect provider* primitive. The domain parameters are defined as follows:

a) *Maximum number of MCS channels that may be in use simultaneously* – This includes channels that are joined by any user, user Ids that have been assigned, and private channels that have been created.

b) *Maximum number of user Ids that may be assigned simultaneously* – This is a sublimit within the constraint of the previous parameter.

c) *Maximum number of token Ids that may be grabbed or inhibited simultaneously*.

d) *Number of data transfer priorities implemented* – An MCS user may still send and receive data with priorities outside the limit. However, such priorities may be treated the same as the lowest priority that is implemented.

e) *Enforced throughput* – This parameter instructs MCS providers to enforce a minimum receiving rate at each MCS attachment and over each downward MCS connection. Violators run the risk of being involuntarily detached or disconnected, respectively.

f) *Maximum height* – This constrains the height of all MCS providers, in particular the Top MCS Provider.

g) *Maximum size of domain MCSPDUs* – Global flow control is based on buffering within an MCS provider. For simplicity, fixed-size buffers are assumed. An MCS provider shall not generate larger MCSPDUs. This constrains the number of parameters that can be packed into a single control operation and suggests where unlimited user data should be segmented.

h) *Protocol version* – This is defined in the MCS protocol specification.

8 MCS channels

Once the domain setup and user attachment is completed, the last step to be performed before data can be exchanged between all the sites in a multipoint fashion is to join the right combination of interaction channels.

Channels in MCS are domain-wide addresses. When a multipoint domain is established, distribution lists can be declared in the form of multicast channels. Every user of a domain can join a channel to receive MCS data units sent to it, and by joining an appropriate combination of channels, a user can choose to receive messages sent to these channels and ignore messages sent to other channels. Application clients subscribe to and leave the desired channels with the *channel join* and *leave* services.

A mechanism for Quality of Service negotiation on a per channel basis is left for further study.

8.1 Multicast channels

Multicast channels are multi-member channels, i.e. channels that can be used to send data to all or a subset of clients of a domain. A multicast channel may represent all active participants in a conference, while another may specify members of a sub-conference. By joining an appropriate combination of channels, a client can elect to receive messages sent to these channels and ignore messages sent to the other channels. For example, all users of a multipoint application may decide to use channel 5 when they wish to receive broadcast data from every member of the domain. Thus every domain member will join channel 5 and send data on this channel when it is to be received by everyone. Another channel may be joined by a subset of domain clients, who will then be able to exchange data on that channel transparent to the rest of domain members.

NOTE – An application client does not have to be joined to a channel in order to send to it, but must be part of it in order to receive data sent to it.

Channel membership is maintained in a distributed fashion, with partial membership information recorded at each level of the hierarchy in a multiple provider domain.

8.2 Single-member channels

Single-member channels are normally used as user identifiers (user Ids) which provide user identification and serve as addresses for point-to-point communication within the multipoint domain. Having such a user Id allows other users to be notified when the user detaches. In order for one

member of a domain to send to another transparent to other sites participating in the same domain, he can send data on channel set to the destination user's Id.

8.3 Private channels

Both send and receive access to individual multicast channels can be controlled by a private channel mechanism. Any user may convene a private channel with *channel convene* which results in him becoming the private channel manager of an available empty multicast channel. The private channel manager can invite users to join the channel using the *channel admit* service or force a user to leave with the *channel expel* service. Users join and leave private channels using the regular *channel join* and *channel leave* services. The private channel manager discontinues a private channel with the *channel disband* service.

8.4 Channel Id numbering

Channel Ids are classified into four types:

– *Static channel Id* – Permanently available, can be joined at will.

– *User Id* – Can be joined only by the MCS attachment that is assigned the MCS user Id.

– *Private channel Id* – Can be joined only by the private channel manager and other authorized user Ids that it explicitly admits.

– *Assigned channel Id* – Created as the result of a *channel join* request for channel Id zero, can be joined at will thereafter, until all joined users leave, at which point it is deleted.

Static channel Ids are numbered 1..1000. All numbers in the static range represent valid channels. Dynamic channel Ids (user Ids, private channel Ids, and assigned channel Ids) are numbered 1001..65535. At any given time, most numbers in the dynamic range will represent non-existent channels (because they were never created or already deleted).

The number of channel Ids in use at once is limited by an MCS domain parameter. Static channel Ids are in use when they are joined by some user. Dynamic channel Ids are in use if they have been created and not yet deleted. MCS primitives (*channel join, attach user, channel convene*) will fail if they threaten to exceed the limit on number of channel Ids in use.

9 MCS data transfer

Once the participating providers have connected and bound to the common domain and users have attached to the common domain and joined the right combination of channels, users are ready to exchange data in a multipoint fashion.

The *send data* and *uniformly sequenced send data* services provide the actual transfer of data. Each send data unit can be delivered (multicast) to multiple sites. Uniform data sequencing, or delivery at each site of identical sequences of data units, is provided with the *uniformly sequenced send data* service.

9.1 Simple Send

The *send data* service provides the "one-to-many" communication, which includes point-to-point as a particular case. Sending to a channel set to a remote user Id will provide the actual point-to-point message transmission. Since any sender can send a service data unit to any channel, the many-to-one and many-to-many operations are also supported. Simple send service data units from different senders may arrive in a different sequence at each site, since they are delivered using the most direct route, up and down the tree of MCS providers. Some sequencing occurs even without using the

uniformly sequenced send data: The sequencing of service data units sent from one sender on one channel at one priority is maintained identically at all receivers. The sender of a simple send request will NOT receive a send indication if it is a member of the destination channel.

MCS service data units are allowed an unlimited size, however, MCS interface data units within a given system may have a maximum size. It is not desirable to split and reassemble long messages transparently within MCS for two reasons. First, the information may be of near real-time nature and may have to be acted upon immediately. Second, since data units from multiple sources are interleaved at different levels in the network, it would not be possible to reconstruct a message from one source without blocking data units from all the other sources. In the application, however, it is possible to analyse incoming data units to process data units that contain real-time information (such as annotations or pen movements) immediately, while reconstructing a long message received from one source (such as an image or a file).

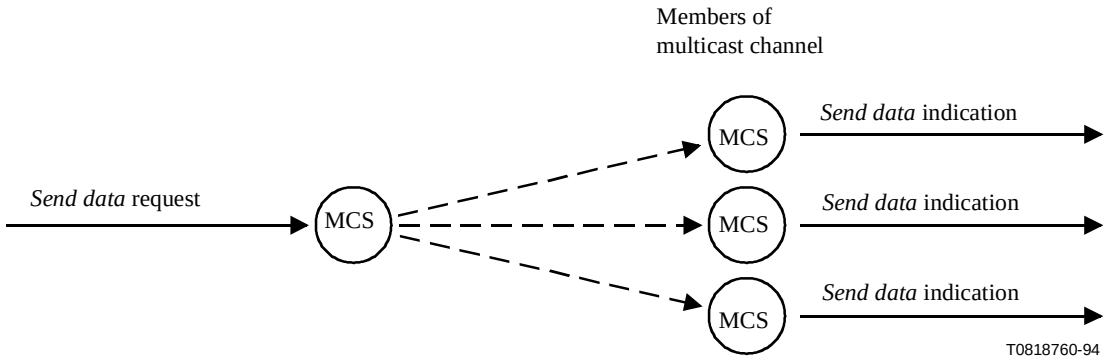


Figure 5/T.122 – Send data to all the members of a multicast channel

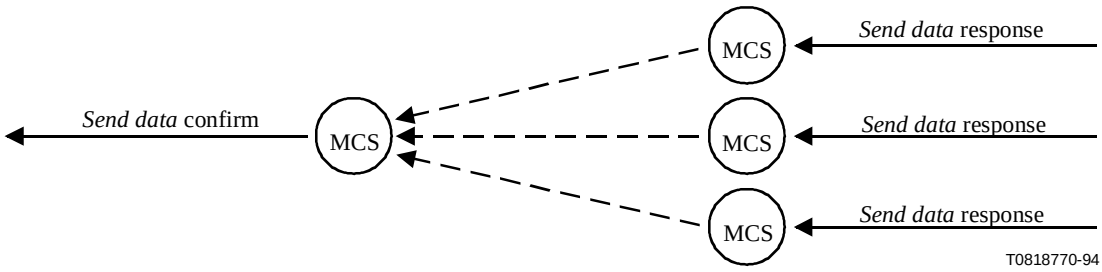
MCS provides sequenced messaging to multiple destinations, as specified by the membership of the destination channel. To adapt the sending rate to the receiving rate, MCS must combine flow control on the individual connections into a global domain-wide flow control. The throughput of a domain may be limited by its slowest receiver.

9.2 Uniformly sequenced data transfer

Uniform sequencing is necessary when data is being sent simultaneously from several sites, but must be received in the same sequence by all receivers. The *uniformly sequenced send data* offers this service. All the *uniformly sequenced send data* requests are routed to the Top MCS provider and from there dispatched in the same order to all the receiving sites, including the sender if it is a member of the destination channel.

9.3 Send data with responses (For further study)

This is a simple form of "many-to-one" communication that occurs as an atomic response to a send. When requested by the sender, each receiver must generate a short response. Responses from all receivers are gathered at each level and travel back along the distribution path of the original message. The aggregate response is then delivered to the sender. This form of response can be used either for feedback on delivery of data or as atomic feedback to a simple query. If a more detailed response is needed, any receiver can send a message to the originator, using the source user Id of the received message.



NOTE – A mechanism must be defined to handle receivers that do not generate the requested response within a reasonable time.

Figure 6/T.122 – Responses requested by a multicast Send

9.4 Unreliable data transfer

By default, all MCS data transfer is fully reliable (i.e. guaranteed delivery). However, there are classes of applications where the occasional loss of data is acceptable and even preferable to the performance loss incurred by recovering lost data with a guaranteed delivery scheme. These applications may choose to use unreliable (i.e. non-guaranteed delivery) data transfer. In this case, received data will be delivered by MCS in the order it was originally sent, but gaps may occur in the application's data stream.

10 Token management

Tokens provide a means to implement exclusive access. For example, to ensure in a multipoint application using resources that one and only one site holds a given resource at a given time, a token can be associated with every resource. When a site wishes to use a specific resource, it must ask for its corresponding token, which will be granted only if no one else is holding it.

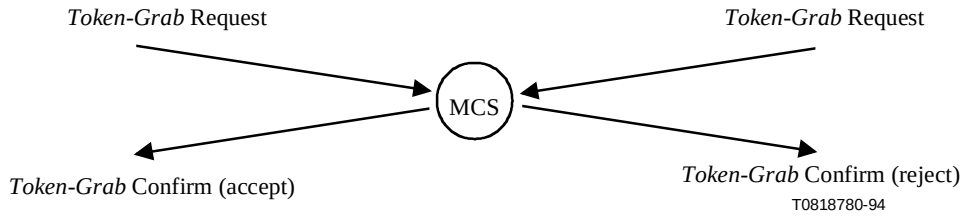


Figure 7/T.122 – Contention resolution: Two sites simultaneously asked for a token, only one site grabbed it

10.1 Exclusive event control and transfer

The *token grab* service allows one user to exclusively hold a given token. The user defines the significance of this token to his application. Other users may use the *token test* service to determine the status of a token at any time and may request the token from the holder with the *token please* service. The token holder may transfer control of a token to another specified user with the *token give* service or return a token to a generally available status with the *token release* service.

10.2 Event coordination

A single token may be used to coordinate a multiple user event by using the *token inhibit* service. Users can independently inhibit and release the same token. For example, if it was desired to know when all users have completed reception and processing of a bulk file transfer, all receiving users would inhibit the same token and each individual user would release the token when it had completed the proscribed process. Any user could test the token at will to determine if the token is free which means all users have completed processing.

10.3 Token Id Numbering

Token Ids are numbered 1..65535. All numbers in the range are valid token Ids. The number of token Ids in use at once is limited by an MCS domain parameter. Token Ids are in use when they are grabbed or inhibited by some user. MCS primitives (*token grab*, *token inhibit*) will fail if they threaten to exceed the limit on number of token Ids in use.

11 Introduction to MCS service primitives

The MCS offers the following service primitives to the application client:

11.1 MCS domain management primitives

– MCS-CONNECT-PROVIDER request, indication, response, confirm
– MCS-DISCONNECT-PROVIDER request, indication
– MCS-ATTACH-USER request, confirm
– MCS-DETACH-USER request, indication
– MCS-DOMAIN-PARAMETERS request, indication, confirm

11.2 MCS Channel Management primitives

– MCS-CHANNEL-JOIN request, confirm
– MCS-CHANNEL-LEAVE request, indication
– MCS-CHANNEL-CONVENE request, confirm
– MCS-CHANNEL-DISBAND request, indication
– MCS-CHANNEL-ADMIT request, indication
– MCS-CHANNEL-EXPEL request, indication

11.3 MCS data transfer primitives

– MCS-SEND-DATA request, indication
– MCS-UNIFORM-SEND-DATA request, indication

11.4 MCS token management primitives

– MCS-TOKEN-GRAB request, confirm
– MCS-TOKEN-INHIBIT request, confirm
– MCS-TOKEN-GIVE request, indication, response, confirm
– MCS-TOKEN-PLEASE request, indication
– MCS-TOKEN-RELEASE request, confirm
– MCS-TOKEN-TEST request, confirm

12 MCS domain management primitives

12.1 MCS-CONNECT-PROVIDER

12.1.1 Function

This primitive establishes an MCS connection through the Control-MCSAP of an MCS provider. It is a local matter how a specific MCS connection end point is identified within the Control-MCSAP. This primitive is confirmed and depends on a response by the MCS user at the Control-MCSAP of the called MCS provider.

The remote site usually passes the indication through a Control-SAP to an application controller. If no control application has attached to the Control-SAP at the remote MCS provider, it is a site configuration decision whether or not all incoming connection requests are to be accepted or rejected.

At the establishment of an MCS connection, two domains that were previously separate are joined into one. Conflicts will arise if the same MCS user Ids are assigned in both. This is resolved by delivering MCS-DETACH-USER indications to selected MCS attachments in the subordinate domain before concluding the MCS Connection. Channel and token Id conflicts are resolved by delivering MCS-CHANNEL-LEAVE and MCS-CHANNEL-DISBAND indications and MCS-DETACH-USER indications to selected MCS attachments in the subordinate domain before concluding the MCS connection.

12.1.2 Types of primitives and their parameters

Table 1/T.122 – MCS-CONNECT-PROVIDER

Primitive/ Parameter	Request	Indication	Response	Confirm
Calling address	M	M(=)		
Calling domain selector	M	M(=)		
Called address	M	M		
Called domain selector	M	M		
Upward/Downward flag	M	M(=)		
Domain parameters	M	M	M	M(=)
Quality of Service	M	M	M	M(=)
Result			M	M(=)
User data	U	C(=)	U	C(=)
NOTE – See 5.1 for the key to this table.				

– *Calling address, Called address* – Used to establish transport connections between MCS providers.

– *Calling domain selector, Called domain selector* – Identifies a domain hosted by the respective MCS provider. An MCS connection binds to the calling and called selectors, joining them into the same domain.

– *Upward/Downward flag* – Designates the called MCS provider as hierarchically superior/subordinate to the calling MCS provider.

– *Domain parameters* – The number of data transfer priorities implemented and, for each, the throughput and transit delay in each direction; also, the number of user Ids, channels and tokens available to MCS users. Data transfer priorities are numbered 0..3 respectively for top, high, medium, and low. If fewer than four priorities are implemented in an MCS domain, lower priority data is transferred at the lowest priority implemented. However, the priority value specified in a data request is indicated unchanged to subsequent receivers. As well, the protocol version, the minimum throughput enforced, the maximum height of all MCS providers, and the maximum size of domain MCSPDUs are all specified as domain parameters.

  These domain parameters are negotiated between the first two MCS providers that connect to a domain. With the exception of protocol version, MCS providers joining a domain after the initial two have no say in determining domain parameters. See 7.3.

– *Quality of Service* – Transport layer Quality of Service. Quality of Service can vary from one MCS connection to another.

– *Result* – Successful, or unsuccessful because of: congested, domain not hierarchical, no such domain, domain parameters unacceptable, unspecified failure, or user-rejected.

– *User Data* – Of unlimited size. MCS users at the Control-MCSAPs of the MCS providers may use this data to align presentation contexts, to authenticate each other, or for some other purpose.

12.1.3 Sequence of primitives

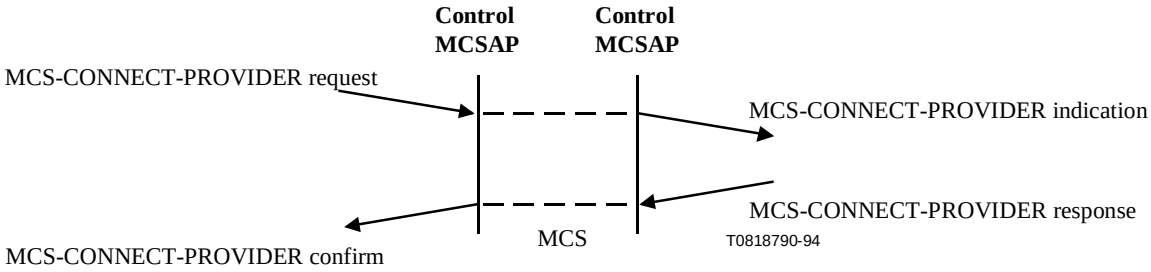


Figure 8/T.122 – MCS-CONNECT-PROVIDER (User accept/reject)

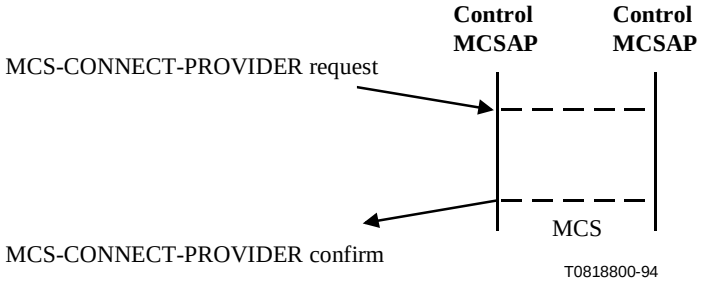


Figure 9/T.122 – MCS-CONNECT-PROVIDER (Provider reject)

12.2 MCS-DISCONNECT-PROVIDER

12.2.1 Function

This primitive releases an MCS connection that was established previously by invocation of MCS-CONNECT-PROVIDER. It is a local matter how a specific MCS connection end point is identified within the Control-MCSAP of an MCS provider. If user-requested, an indication is delivered through the Control-MCSAP of the MCS provider at the other end of the MCS connection. If provider-initiated, an indication is delivered at both ends.

At the release of an MCS connection, the domain that contained it is split into two. MCS users attached to the Top MCS Provider portion will receive MCS-DETACH-USER indications for those in the opposite portion. If a more severe response is called for, it is the duty of MCS users at the affected Control-MCSAPs to decide. They may inform and direct other MCS users via MCS data transfer or by local means. It is for future study whether the users in the disconnecting bottom portion establish their own domain or not.

12.2.2 Types of primitives and their parameters

Table 2/T.122 – MCS-DISCONNECT-PROVIDER

Primitive/ Parameter	Request	Indication
Reason		M
NOTE – See 5.1 for the key to this table.		

– *Reason* – Domain disconnected, domain not hierarchical, domain parameters unacceptable, provider-initiated, unspecified, user-requested.

12.2.3 Sequence of primitives

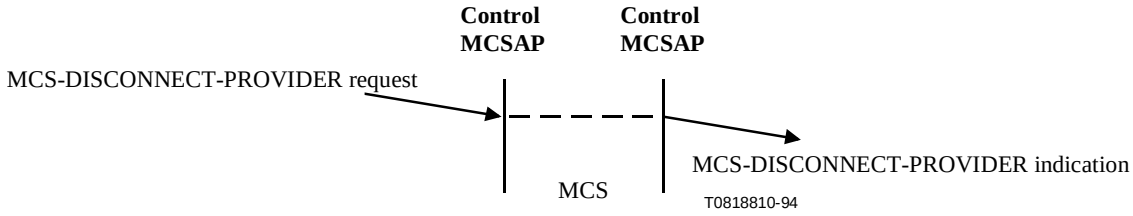


Figure 10/T.122 – MCS-DISCONNECT-PROVIDER (User-initiated)

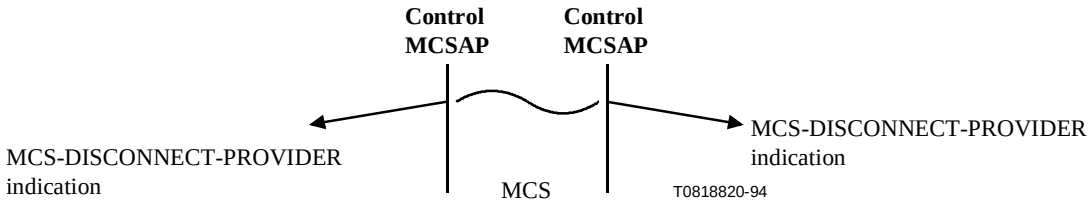


Figure 11/T.122 – MCS-DISCONNECT-PROVIDER (Provider-initiated)

12.3 MCS-ATTACH-USER

12.3.1 Function

This primitive creates an MCS attachment through an MCS SAP to a domain hosted by the MCS provider. A result is confirmed to the requester. If the request is accepted, a user Id is assigned.

All subsequent MCS primitives are invoked in the context of some MCS attachment. It is a local matter how a specific MCS attachment is identified within an MCS SAP. Note that domain selector and user Id suffice, but other means may be preferred.

12.3.2 Types of primitives and their parameters

Table 3/T.122 – MCS-ATTACH-USER

Primitive/ Parameter	Request	Confirm
Domain selector	M	
Result		M
User Id		C
NOTE – See 5.1 for the key to this table.		

– *Domain selector* – Identifies a domain hosted by the MCS provider.

– *Result* – Successful, or unsuccessful because: congested, domain disconnected, no such domain, too many channels, too many users, unspecified failure.

– *User Id* – Guaranteed unique within the MCS domain. Its value is drawn from the space of MCS channel numbers. To communicate in point-to-point fashion, an MCS user must join the assigned channel as a receiver and must instruct other MCS users to send to it. Channel numbers assigned as user Ids are disjoint from channel numbers representing multicast distribution lists.

12.3.3 Sequence of primitives

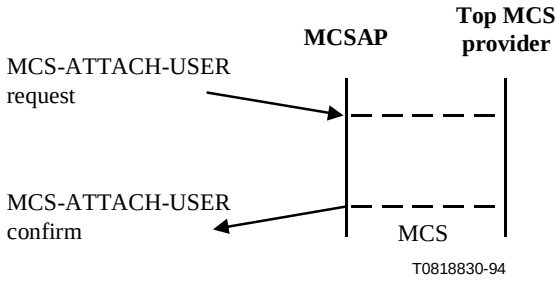


Figure 12/T.122 – MCS-ATTACH-USER

12.4 MCS-DETACH-USER

12.4.1 Function

This primitive deletes an MCS attachment that was created previously by invocation of MCS-ATTACH-USER. It is a local matter how a specific MCS attachment is identified within an MCS SAP. This primitive may be requested by a user or initiated by a provider. It delivers an indication at

every other MCS attachment to the same domain. If provider-initiated, an indication is also delivered at the deleted attachment.

12.4.2 Types of primitives and their parameters

Table 4/T.122 – MCS-DETACH-USER

Primitive/ Parameter	Request	Indication
Reason		M
User Id		M
NOTE – See 5.1 for the key to this table		

– *Reason* – Channel purged, domain disconnected, provider-initiated, token purged (Until there is a TOKEN-RELEASE indication, issuing an MCS-DETACH-USER indication is the only way for the MCS provider to cause a token to be purged. This may be necessary during domain merging.), unspecified, user-requested.

– *User Id* – Assigned by MCS-ATTACH-USER.

12.4.3 Sequence of primitives

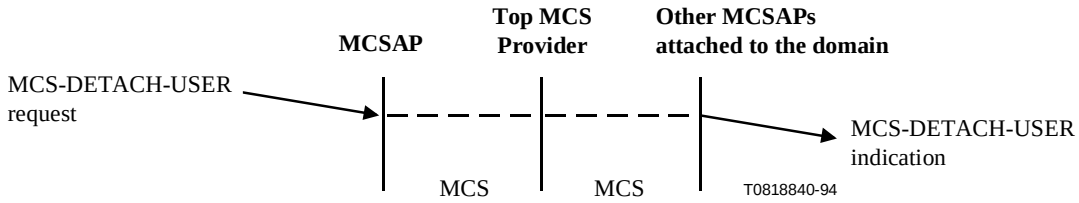


Figure 13/T.122 – MCS-DETACH-USER (User-initiated)

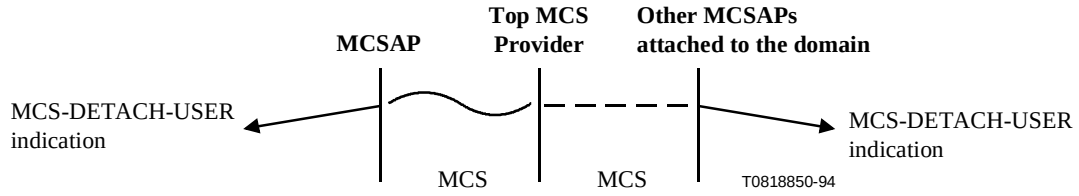


Figure 14/T.122 – MCS-DETACH-USER (MCS provider-initiated)

12.5 MCS-DOMAIN-PARAMETERS

12.5.1 Function

This primitive provides an MCS user with the parameter values associated with a specified domain. A confirm is received in response to a user-initiated request. An indication is provider-initiated, and is received immediately after successfully attaching to a domain.

12.5.2 Types of primitives and their parameters

Table 5/T.122 – MCS-DOMAIN-PARAMETERS

Primitive/ Parameter	Request	Indication	Confirm
Domain parameters		M	M
Unreliable data is supported		M	M
Maximum user data length		M	M
Maximum admit/expel user Ids		M	M
NOTE – See 5.1 for the key to this table.			

– *Domain parameter*s – The domain parameter constraints used by a domain.

– *Unreliable data is supported* – A Boolean value that indicates if unreliable data transfer is supported by this MCS provider for this domain.

– *Maximum user data length* – The length, in bytes, of the greatest user data field which *MCS-SEND-DATA* request or *MCS-UNIFORM-SEND-DATA* request can send. Fully reliable data larger than this value will be segmented by MCS. Unreliable data larger than this value is not permitted.

– *Maximum admit/expel user Ids* – The number of user IDs which can be listed in an *MCS-CHANNEL-ADMIT* request or *MCS-CHANNEL-EXPEL* request.

12.5.3 Sequence of primitives

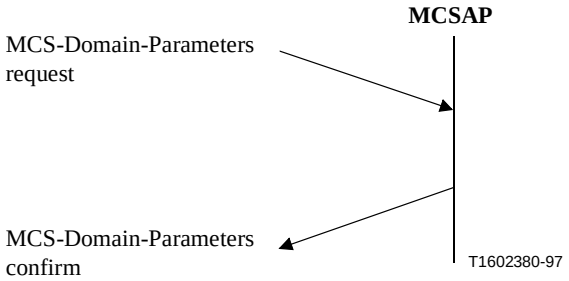


Figure 15/T.122 – MCS-DOMAIN-PARAMETERS request-confirm

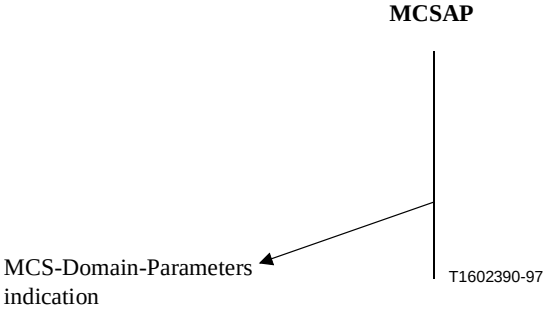


Figure 16/T.122 – MCS-DOMAIN-PARAMETERS indication

13 MCS channel management primitives

13.1 MCS-CHANNEL-JOIN

13.1.1 Function

The MCS-CHANNEL-JOIN service is used by an application client to join an appropriate channel whose use is defined by the application. This is a prerequisite for receiving data sent to the channel.

13.1.2 Types of primitives and their parameters

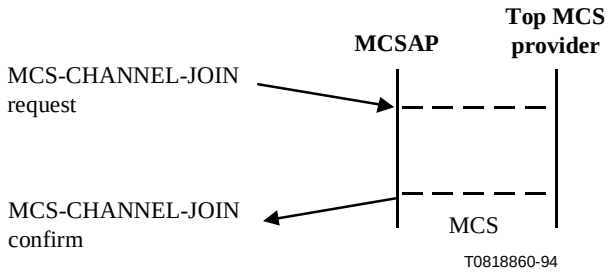
Table 6/T.122 – MCS-CHANNEL-JOIN

Primitive/ Parameter	Request	Confirm
Channel to join	M	C
Result		M
NOTE – See 5.1 for the key to this table.		

– *Channel to join* – Identifies the channel to join in the request, and the actual channel joined in the confirm. If channel = 0, then a currently empty multicast channel is to be joined. An empty channel is one to which no user is joined.

– *Result* – Indicates whether or not joining the channel was allowed. Its value is one of: successful, or unsuccessful because of: other user Id (this channel is a user Id channel already assigned to another user), no such channel, not admitted to the channel, too many channels.

13.1.3 Sequence of primitives



NOTE – A Join indication primitive issued to the channel manager is for further study.

Figure 17/T.122 – MCS-CHANNEL-JOIN

13.2 MCS-CHANNEL-LEAVE

13.2.1 Function

The MCS-CHANNEL-LEAVE service is used by an application client to leave a previously joined channel and thus stop receiving data sent to that channel. The primitive may be user-initiated (request only) or provider-initiated (indication to affected user only).

13.2.2 Types of primitives and their parameters

Table 7/T.122 – MCS-CHANNEL-LEAVE

Primitive/ Parameter	Request	Indication
Channel to leave	M	M
Reason		M
NOTE – See 5.1 for the key to this table.		

– *Channel to leave* – Identifies the channel to leave.

– *Reason* – Provider-initiated because channel purged.

13.2.3 Sequence of primitives

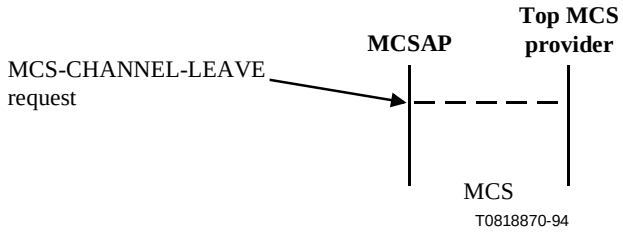


Figure 18/T.122 – MCS-CHANNEL-LEAVE (User-initiated)

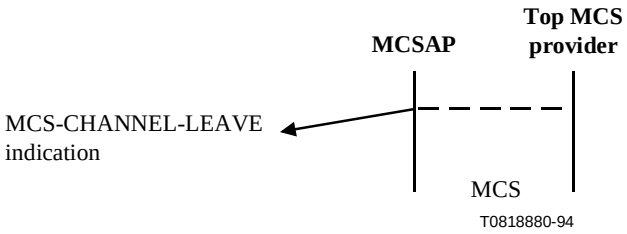


Figure 19/T.122 – MCS-CHANNEL-LEAVE (MCS provider-initiated)

13.3 MCS-CHANNEL-CONVENE

13.3.1 Function

This primitive allocates a new private channel with the requesting user as manager. A result is confirmed; if the request is accepted, a channel number is assigned. The authorized user group initially consists of the manager alone. It is guaranteed that no users are joined to the channel.

13.3.2 Types of primitives and their parameters

Table 8/T.122 – MCS-CHANNEL-CONVENE

Primitive/ Parameter	Request	Confirm
Result		M
Channel		C
NOTE – See 5.1 for the key to this table.		

– *Result* – Successful, or unsuccessful because of: too many channels.

– *Channel* – A private channel number, protected from unauthorized users.

13.3.3 Sequence of primitives

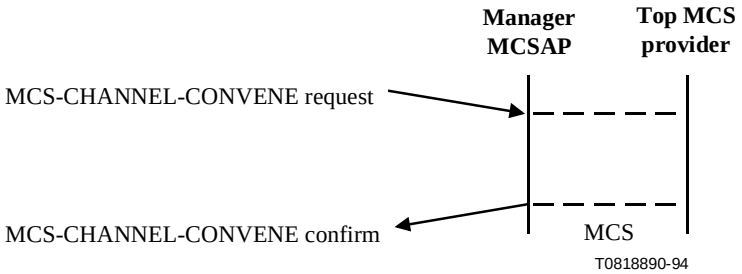


Figure 20/T.122 – MCS-CHANNEL-CONVENE

13.4 MCS-CHANNEL-DISBAND

13.4.1 Function

This primitive deallocates a private channel that was allocated previously by invocation of MCS-CHANNEL-CONVENE. This primitive may be requested by or indicated to the channel's manager. It causes an MCS-CHANNEL-EXPEL indication to be delivered to all members remaining in the authorized user group. Thereafter, the channel number may be recycled for private or public use.

13.4.2 Types of primitives and their parameters

Table 9/T.122 – MCS-CHANNEL-DISBAND

Primitive/ Parameter	Request	Indication
Channel	M	M
Reason		M
NOTE – See 5.1 for the key to this table.		

– *Channel* – A private channel number.

– *Reason* – Provider-initiated because channel purged.

13.4.3 Sequence of primitives

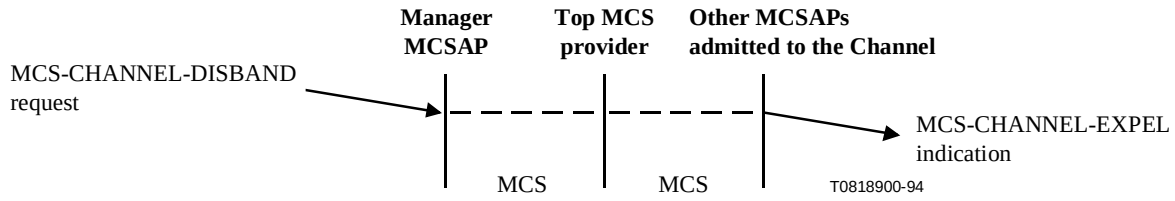


Figure 21/T.122 – MCS-CHANNEL-DISBAND (Channel manager-initiated)

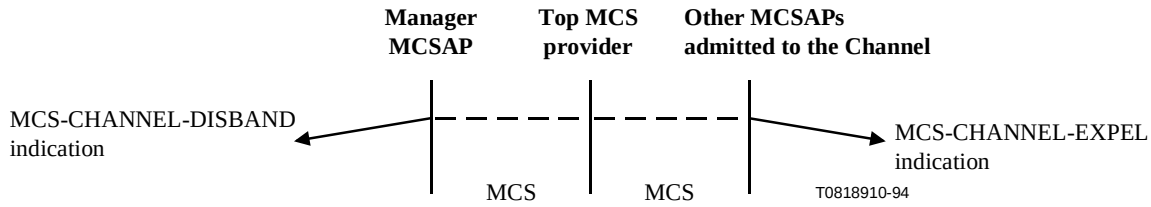


Figure 22/T.122 – MCS-CHANNEL-DISBAND (MCS provider-initiated)

13.5 MCS-CHANNEL-ADMIT

13.5.1 Function

This primitive enlarges the authorized user group of a private channel at the request of its manager. An indication is delivered to the MCS user added. That user may thereafter send data on the channel or join it as a receiver or invoke other MCS primitives.

13.5.2 Types of primitives and their parameters

Table 10/T.122 – MCS-CHANNEL-ADMIT

Primitive/ Parameter	Request	Indication
Channel	M	M(=)
Manager user Id		M
List of user Ids	M	
NOTE – See 5.1 for the key to this table.		

– *Channel* – A private channel number.

– *Manager User Id* – The user Id of the channel manager.

– *List of User Ids* – The list of users to be added to the authorized group.

13.5.3 Sequence of primitives

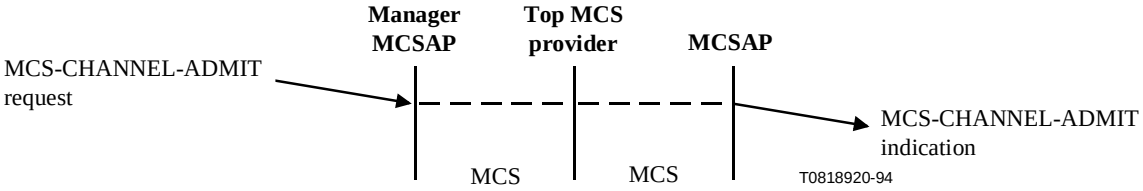


Figure 23/T.122 – MCS-CHANNEL-ADMIT

13.6 MCS-CHANNEL-EXPEL

13.6.1 Function

This primitive shrinks the authorized user group of a private channel. It may be requested by the channel manager or initiated by the MCS provider. An indication is delivered to the MCS user deleted. If the deleted user is joined to the channel as a receiver the expel also has the effect of a channel leave indication.

13.6.2 Types of primitives and their parameters

Table 11/T.122 – MCS-CHANNEL-EXPEL

Primitive/ Parameter	Request	Indication
Channel	M	M (=)
List of user Ids	M	
Reason		M
NOTE – See 5.1 for the key to this table.		

– *Channel* – A private channel number.

– *List of user Ids* – The list of users to be expelled from the authorized group.

– *Reason* – Channel disbanded, channel purged, user-requested.

13.6.3 Sequence of primitives

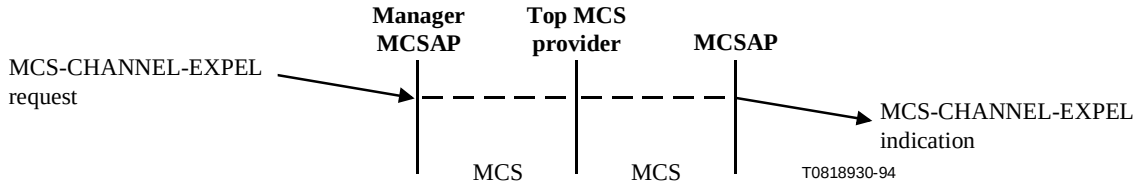


Figure 24/T.122 – MCS-CHANNEL-EXPEL

14 MCS data transfer primitives

14.1 MCS-SEND-DATA

14.1.1 Function

The MCS-SEND-DATA service is used to transmit data to other members of a domain.

If the sender is a member of the destination channel, it will not receive its own data indications. However, it will receive data indications from other sources addressed to that channel.

If multiple clients send data to the same channel, different receivers may receive data from different senders in a different order. If uniform sequencing is needed for some data, all senders should send that data using the MCS-UNIFORMLY-SEQUENCED-SEND-DATA services described later.

Fully reliable data from each sender sent at the same priority on the same channel arrive at a given receiver in the same order as it was sent but may have other sender data interleaved differently. Unreliable data will be similarly ordered at the receiver, but there may be data packets missing from a sender's data stream.

14.1.2 Types of primitives and their parameters

Table 12/T.122 – MCS-SEND-DATA

Primitive/ Parameter	Request	Indication
Reliability	M	M(=)
Priority	M	M(=)
Channel Id	M	M(=)
Sender user Id		M
Segmentation		M
Total data size		C
Data	M	M(=)
NOTE – See 5.1 for the key to this table.		

– *Reliability* – Indicates whether the data should be sent, or was received, via fully reliable or unreliable transport.

– *Priority* – The number of priority levels implemented is a Quality of Service parameter of the domain.

– *Channel Id* – Indicates which channel to use to send the data.

– *Sender user Id* – Is set by the sender's MCS provider.

– *Segmentation* – The segmentation flags *begin* and *end* permit fully reliable data units to be reassembled by the user. These flags shall be interpreted in the context of MCS-SEND-DATA indications arriving from the same user over the same channel and at the same priority. A stream of fragments to be reassembled may be interleaved with other MCS primitives and data from other users over other channels at other priorities. Unreliable data units will not be segmented by MCS and will not require reassembly by the user.

– *Total data size* – For a given MCS-SEND-DATA indication, if the *begin* segmentation flag is true, and the *end* flag is false, the total size of the segmented user data will be provided.

This parameter will only be present if both the source and sink nodes are members of the V3 summit.

– *Data* – If *Reliability* is fully reliable, the data can be of unlimited size. If *Reliability* is unreliable, the data size may not exceed the *Maximum User Data Length* returned by the MCS-Get-Domain-Parameters primitive. See 12.5.

14.1.3 Sequence of primitives

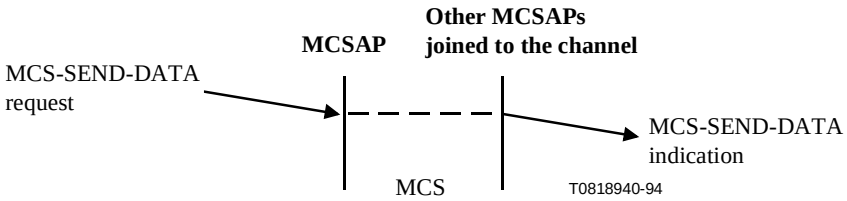


Figure 25/T.122 – MCS-SEND-DATA

14.2 MCS-UNIFORMLY-SEQUENCED-SEND-DATA

14.2.1 Function

The MCS-UNIFORMLY-SEQUENCED-SEND-DATA service is used to transmit data to other members of a domain in a uniformly sequenced manner, i.e. the data will be received in the same sequence by all members of the destination channel. The different data units from the domain clients will be forwarded to the Top MCS Provider, which will send them back to all clients in the same sequence. Note that if unreliable transmission is selected, data packets may be missing from the uniform data stream, and that these missing packets may differ from node to node.

Uniform sequencing of data is guaranteed only for data of the same priority on the same channel.

If the sender is a member of the destination channel, it will receive its own data, in the proper sequence. Therefore, it is provided with the same ordered data stream as other members of the channel.

14.2.2 Types of primitives and their parameters

Table 13/T.122 – MCS-UNIFORM-SEND-DATA

Primitive/ Parameter	Request	Indication
Reliability	M	M(=)
Priority	M	M(=)
Channel Id	M	M(=)
Sender user Id		M
Segmentation		M
Total Data Size		C
Data	M	M(=)
NOTE – See 5.1 for the key to this table.		

– *Reliability* – Indicates whether the data should be sent, or was received, via fully reliable or unreliable transport.

– *Priority* – The number of priority levels implemented is a Quality of Service parameter of the domain.

– *Channel Id* – Indicates which channel to use to send the data.

– *Sender user Id* – Is set by the sender's MCS provider.

– *Segmentation* – The segmentation flags *begin* and *end* permit fully reliable data units to be reassembled by the user. These flags shall be interpreted in the context of MCS-UNIFORM-SEND-DATA indications arriving from the same user over the same channel and at the same priority. A stream of fragments to be reassembled may be interleaved with other MCS primitives and data from other users over other channels at other priorities. Unreliable data units will not be segmented by MCS and will not require reassembly by the user.

– *Total data size* – For a given MCS-UNIFORM-SEND-DATA indication, if the *begin* segmentation flag is true, and the *end* flag is false, the total size of the segmented user data will be provided. This parameter will only be present if both the source and sink nodes are members of the V3 summit.

– *Data* – If *Reliability* is fully reliable, the data can be of unlimited size. If *Reliability* is unreliable, the data size may not exceed the *Maximum User Data Length* returned by the *MCS-GET-DOMAIN-PARAMETERS* primitive. See 12.5.

14.2.3 Sequence of primitives

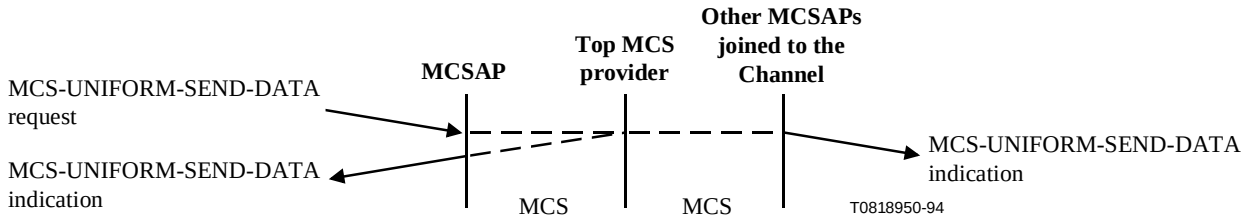


Figure 26/T.122 – MCS-UNIFORMLY-SEQUENCED-SEND-DATA

15 MCS token management primitives

15.1 MCS-TOKEN-GRAB

15.1.1 Function

The MCS-TOKEN-GRAB service is used to take exclusive control of a specific token.

MCS-TOKEN-GRAB will succeed if the requester is the sole inhibitor of the token.

15.1.2 Types of primitives and their parameters

Table 14/T.122 – MCS-TOKEN-GRAB

Primitive/ Parameter	Request	Confirm
Token identifier	M	M(=)
Result		M
NOTE – See 5.1 for the key to this table.		

– *Token Id* – Identifies the token the client wishes to grab.

– *Result* – Indicates whether or not the token grab was allowed. Its value is one of: successful, or unsuccessful because: token not available, too many tokens.

15.1.3 Sequence of primitives

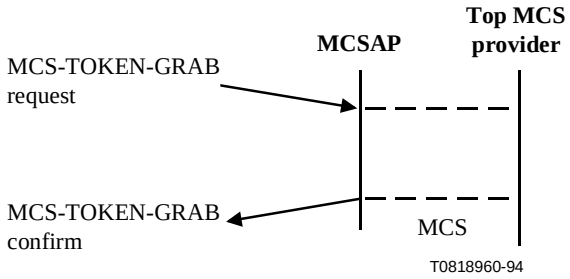


Figure 27/T.122 – MCS-TOKEN-GRAB

15.2 MCS-TOKEN-INHIBIT

15.2.1 Function

The MCS-TOKEN-INHIBIT service is used to take non-exclusive control of a specific token. It is used to prevent someone else from exclusively grabbing the token. Several users could inhibit a token at the same time.

MCS-TOKEN-INHIBIT will succeed if the requester has grabbed the token. The result will be that the token is no longer grabbed and is inhibited instead. Thereafter it may be inhibited by other users too.

15.2.2 Types of primitives and their parameters

Table 15/T.122 – MCS-TOKEN-INHIBIT

Primitive/ Parameter	Request	Confirm
Token identifier	M	M(=)
Result		M
NOTE – See 5.1 for the key to this table.		

– *Token Id* – Identifies the token the client wishes to inhibit.

– *Result* – Indicates whether or not the token inhibit was allowed. Its value is one of: successful, or unsuccessful because: token not available, too many tokens.

15.2.3 Sequence of primitives

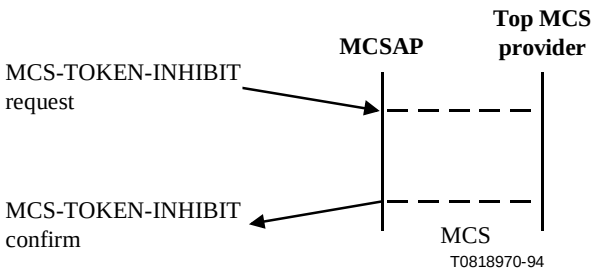


Figure 28/T.122 – MCS-TOKEN-INHIBIT

15.3 MCS-TOKEN-GIVE

15.3.1 Function

The MCS-TOKEN-GIVE service is used by an application client to surrender a token to another application client.

MCS-TOKEN-GIVE will fail if the requester has not grabbed the specified token.

A token being passed between two users and whose possession is not yet resolved will appear to any user requesting MCS-TOKEN-TEST to be grabbed by some other user and not held by the requester. MCS-TOKEN-GRAB and MCS-TOKEN-INHIBIT will fail during the interval, even if requested by one of the two users involved. MCS-TOKEN-RELEASE requested by the giver will succeed, with the result that the token is released if the offer to the recipient is ultimately rejected. MCS-TOKEN-RELEASE by the recipient will have no effect, just like the attempted release of any other token that the requester does not yet possess. During the interval that a token is being passed, any MCS-TOKEN-PLEASE indications that are generated will be delivered to both users involved.

It is possible for the ownership of a passed token to be resolved before an MCS-TOKEN-GIVE response or confirm is issued, i.e. deletion of either the giver or recipient will leave the survivor in sole possession as grabber of the token. At this point, the special effects explained above cease to apply.

15.3.2 Types of primitives and their parameters

Table 16/T.122 – MCS-TOKEN-GIVE

Primitive/ Parameter	Request	Indication	Response	Confirm
User Id giving token		M		
User Id to receive token	M			
Token Id	M	M(=)		M(=)
Result			M	M(=)
NOTE – See 5.1 for the key to this table.				

– *User Id giving token* – The user Id giving the token. It is set by the user's MCS provider.

– *User Id to receive token* – The user Id to be receiving the token.

– *Token identifier* – Is the token to be given.

– *Result* – Indicates whether or not the token give was successful. Its value is one of: successful, or unsuccessful because: domain merging, no such user, token not possessed, user rejected.

15.3.3 Sequence of primitives

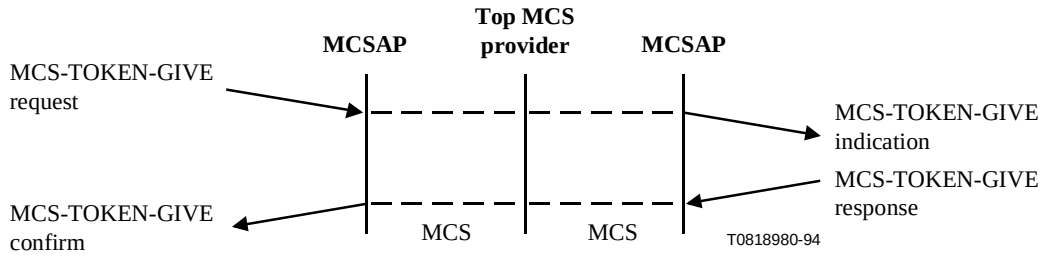


Figure 29/T.122 – MCS-TOKEN-GIVE

15.4 MCS-TOKEN-PLEASE

15.4.1 Function

The MCS-TOKEN-PLEASE service is used by an application client to request a token from the current possessor(s) of the token. A token may be inhibited by several users, or it may be grabbed by one user. In any case, MCS-TOKEN-PLEASE indications are delivered to each user possessing the token.

15.4.2 Types of primitives and their parameters

Table 17/T.122 – MCS-TOKEN-PLEASE

Primitive/ Parameter	Request	Indication
User Id requesting token		M
Token Id	M	M(=)
NOTE – See 5.1 for the key to this table.		

– *User Id requesting token* – Is the user Id requesting the token. It is set by the user's MCS provider.

– *Token Id* – Is the token being requested.

15.4.3 Sequence of primitives

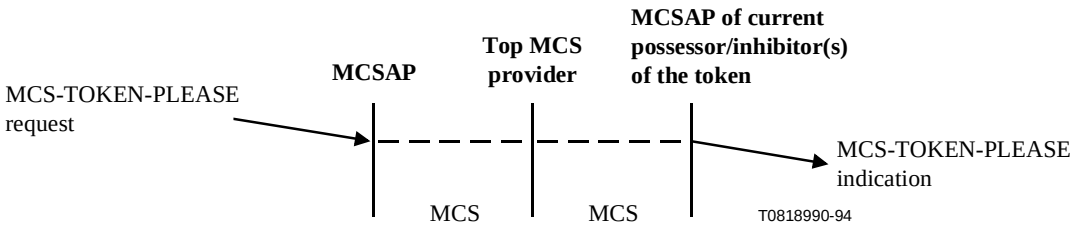


Figure 30/T.122 – MCS-TOKEN-PLEASE

Response and Confirm primitives are left out for further study.

15.5 MCS-TOKEN-RELEASE

15.5.1 Function

The MCS-TOKEN-RELEASE service is used to free up a previously grabbed/inhibited token.

15.5.2 Types of primitives and their parameters

Table 18/T.122 – MCS-TOKEN-RELEASE

Primitive/ Parameter	Request	Confirm
Token Identifier	M	M(=)
Result		M
NOTE – See 5.1 for the key to this table.		

– *Token Id* – The token to release.

– *Result* – Successful, or unsuccessful because: token not possessed.

15.5.3 Sequence of primitives

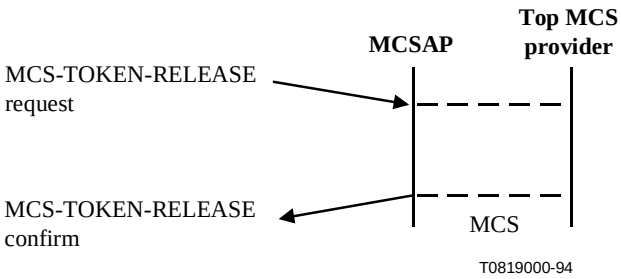


Figure 31/T.122 – MCS-TOKEN-RELEASE (MCS user-initiated)

15.6 MCS-TOKEN-TEST

15.6.1 Function

The MCS-TOKEN-TEST service is used to check whether a token is available.

15.6.2 Types of primitives and their parameters

Table 19/T.122 – MCS-TOKEN-TEST

Primitive/ Parameter	Request	Confirm
Token identifier	M	M(=)
Token status		M
NOTE – See 5.1 for the key to this table.		

– *Token Id* – Is the token to be tested for its status.

– *Token status* – Is one of free, grabbed, grabbed and held by the testing user, inhibited, inhibited and inhibited by the testing user, token does not exist.

15.6.3 Sequence of primitives

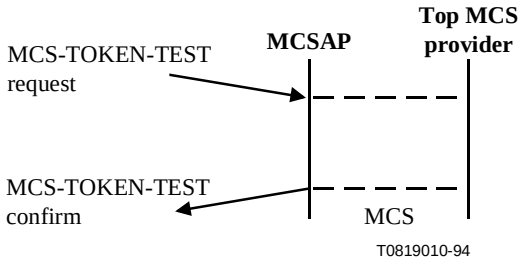


Figure 32/T.122 – MCS-TOKEN-TEST

ANNEX A

Domain establishment, data transfer and release phases examples

A.1 MCS domain establishment phase

In the domain establishment phase, the application using the MCS-CONNECT-PROVIDER asks the MCS provider to establish an MCS connection to a specific MCS provider and bind that connection to a specific domain. User applications then attach themselves to this domain using the MCS-ATTACH-USER primitive. Then, the proper channels are joined to be able to receive the appropriate data (using the MCS-CHANNEL-JOIN, MCS-CHANNEL-CONVENE, MCS-CHANNEL-ADMIT primitives).

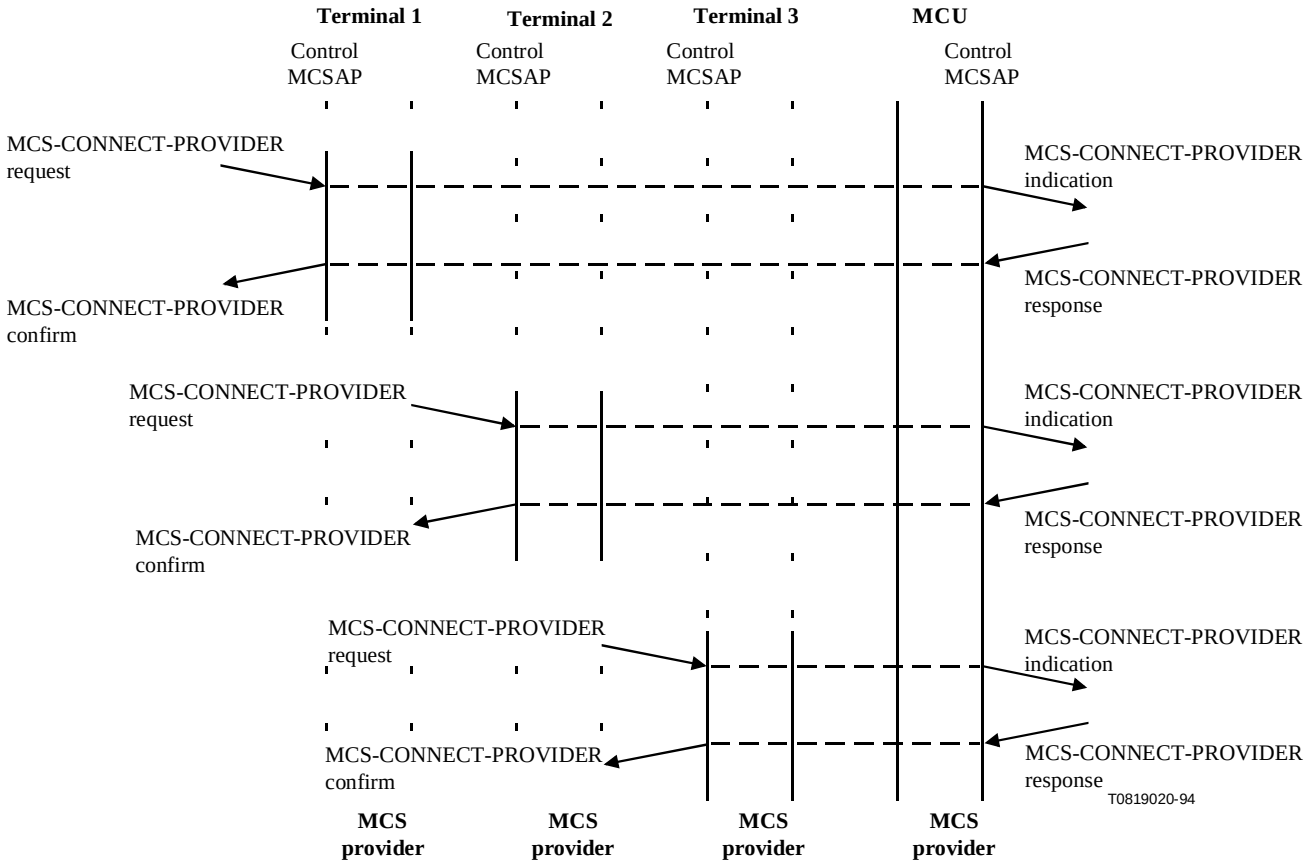


Figure A.1/T.122 – Establishing MCS connections

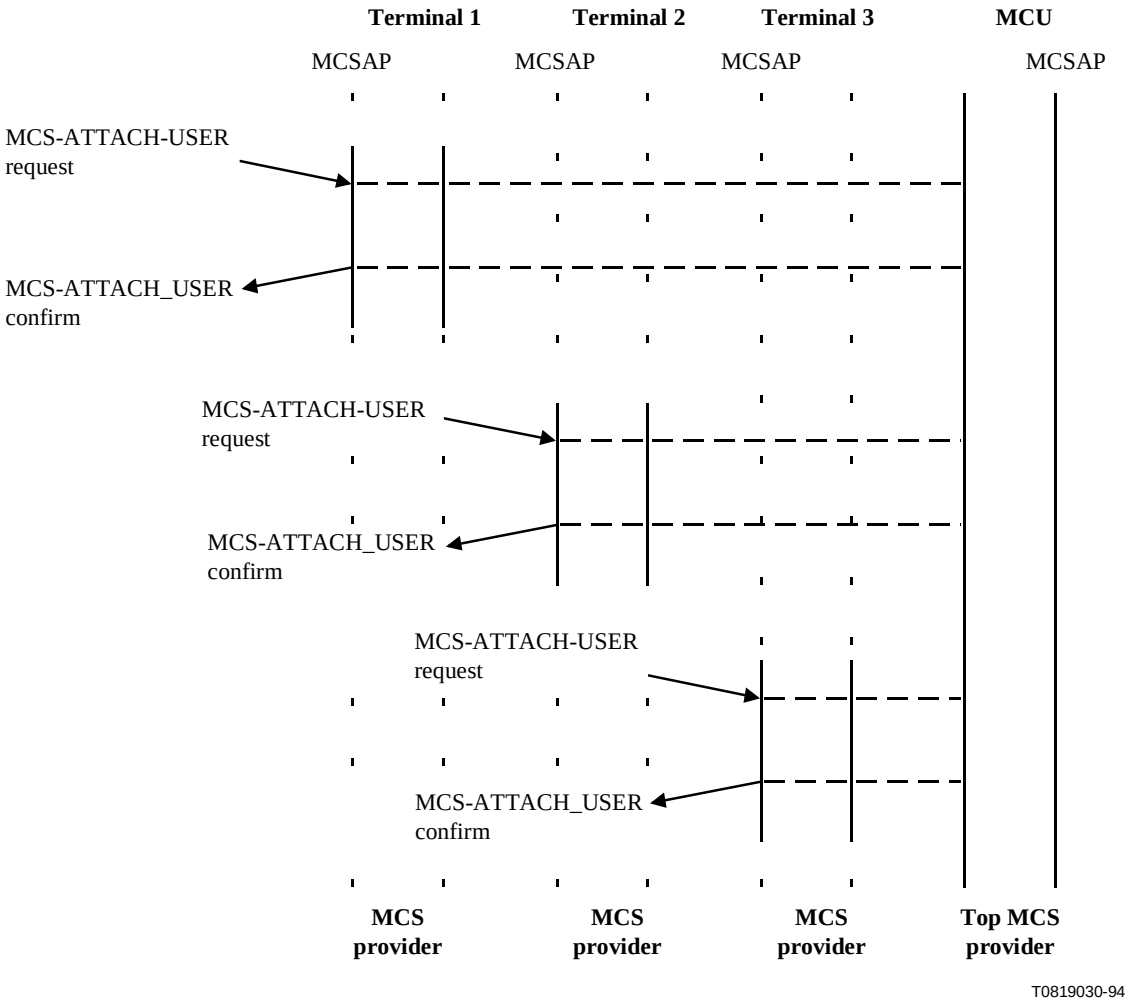


Figure A.2/T.122 – MCS domain establishment

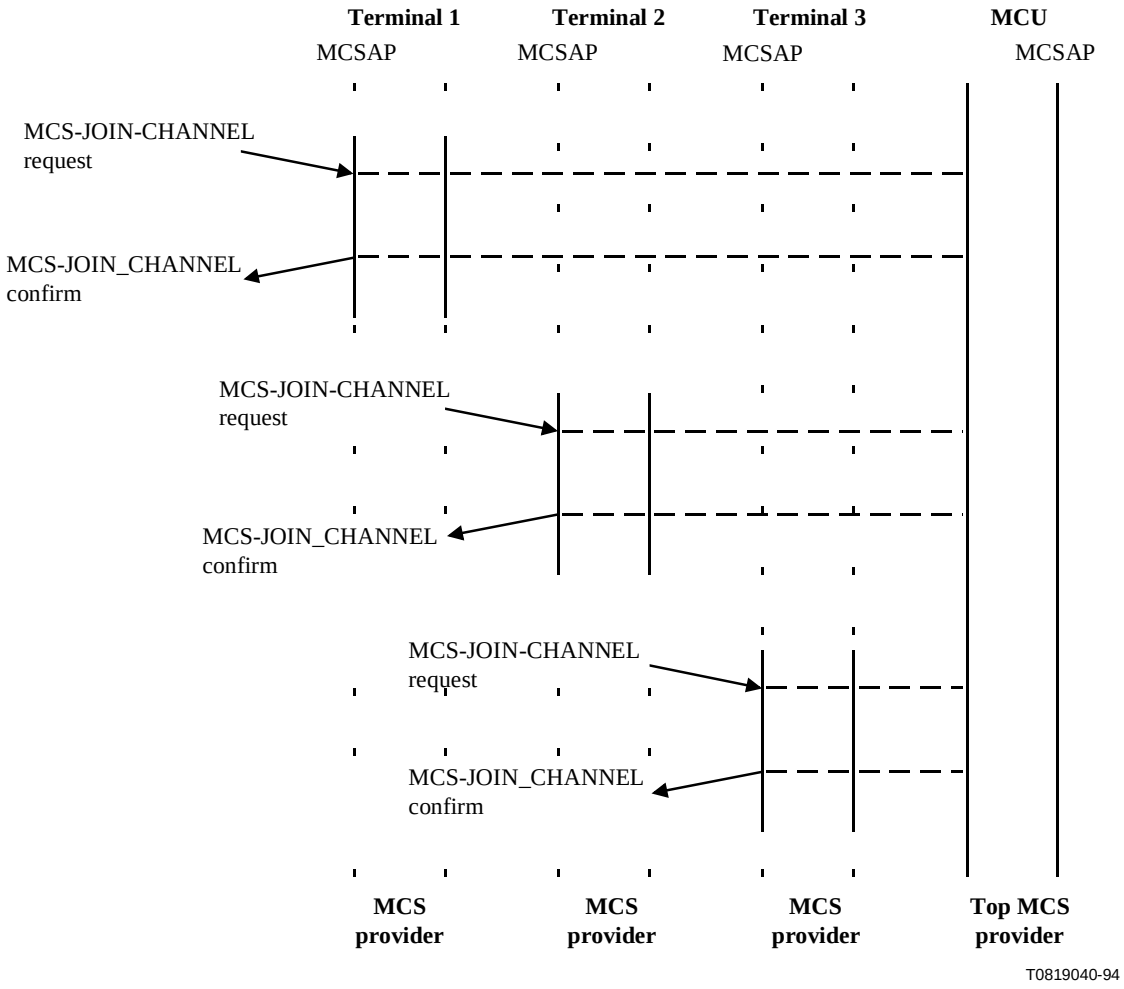


Figure A.3/T.122 – Users joining a multicast public channel: All request the same channel number

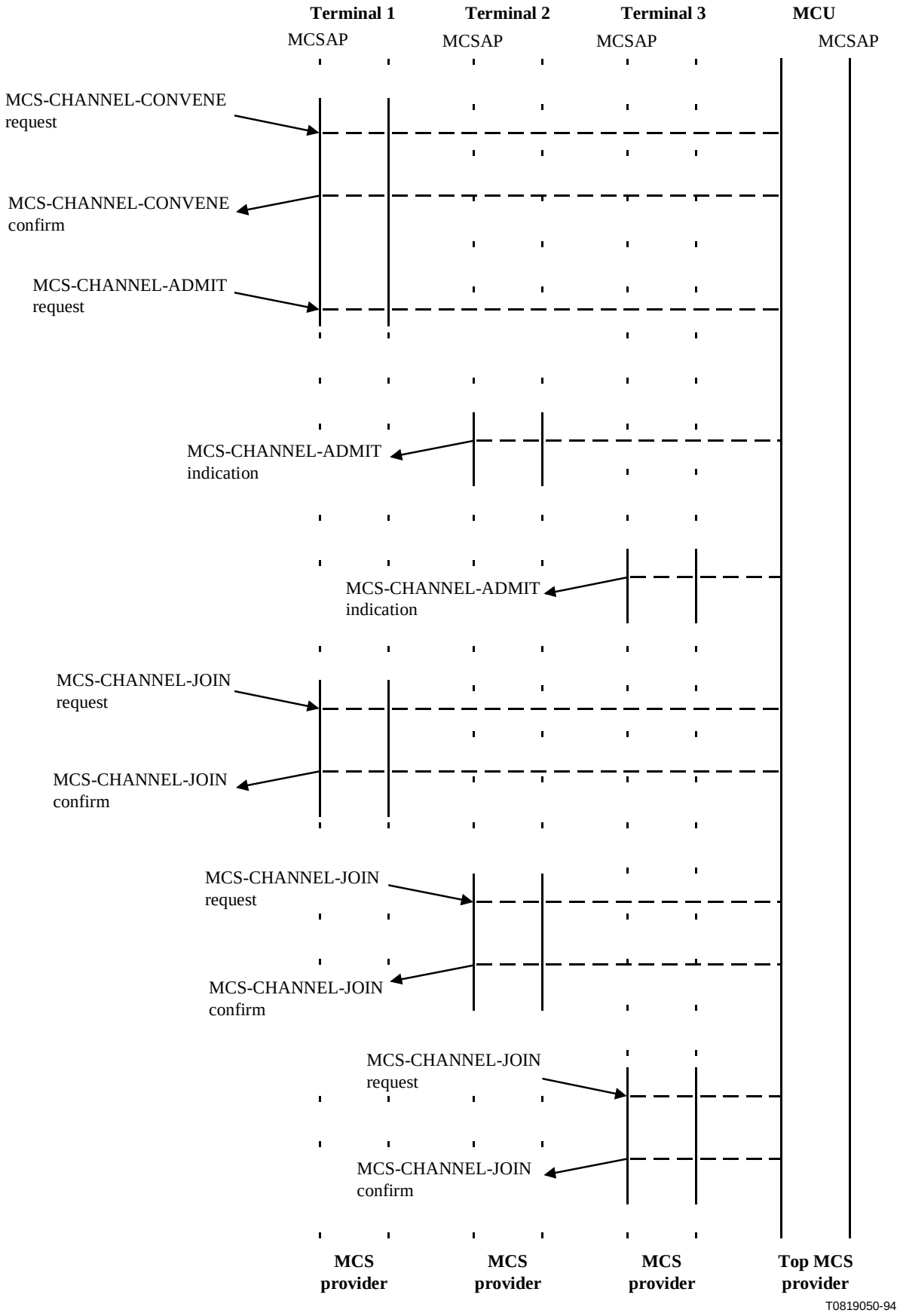


Figure A.4/T.122 – Manager establishes a private channel and then the users join it

A.2 MCS data transfer phase

The data transfer phase comprises use of the MCS-SEND-DATA and the MCS-UNIFORMLY-SEQUENCED-SEND-DATA primitives. Token operations can be performed using MCS-TOKEN-OPERATIONS primitives.

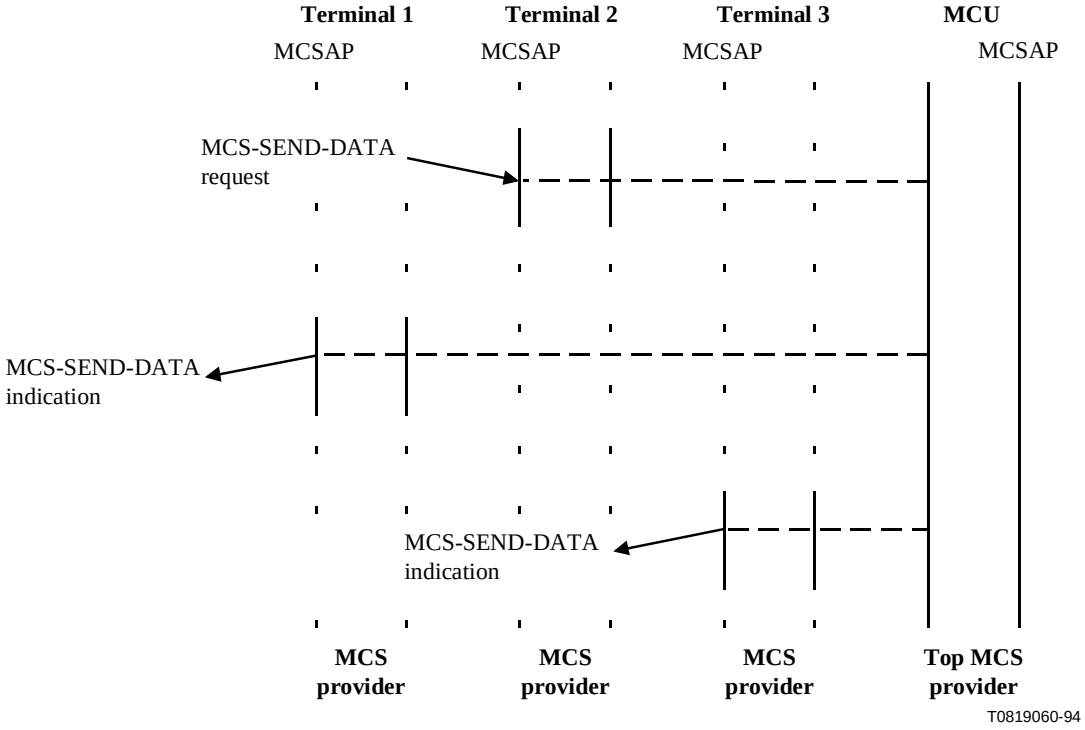


Figure A.5/T.122 – Sending data on a channel

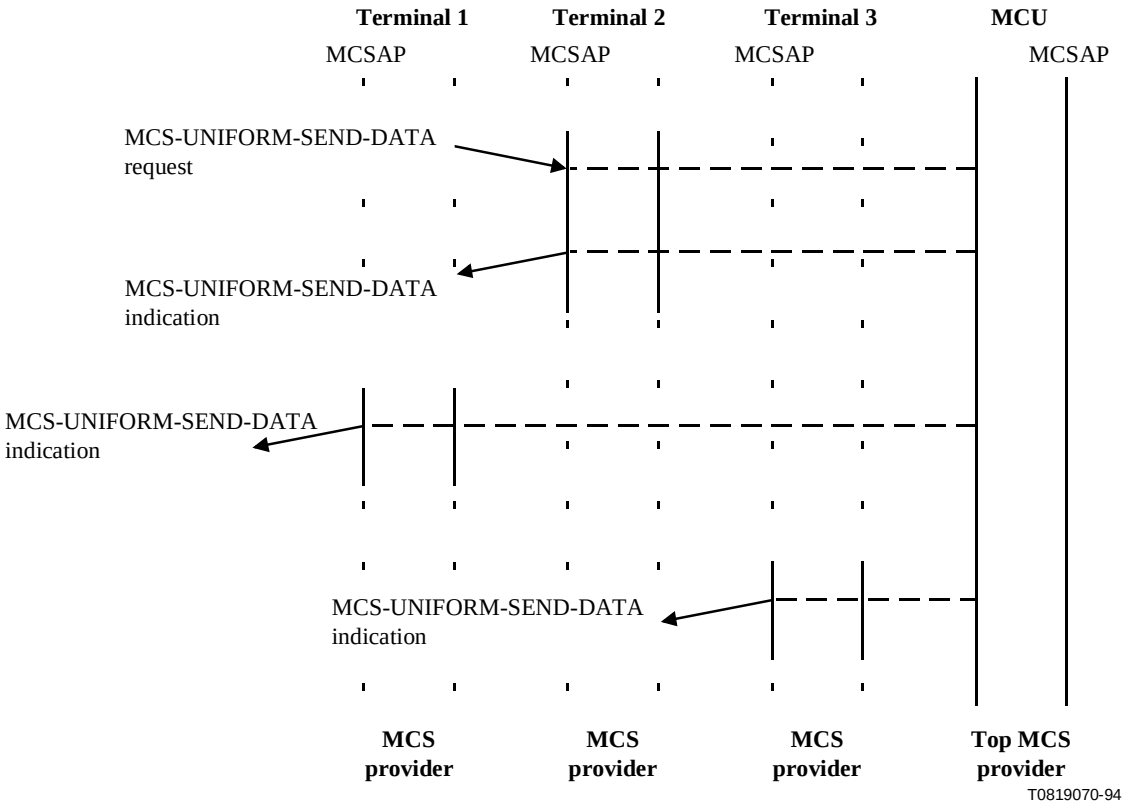


Figure A.6/T.122 – Uniformly sequenced send data

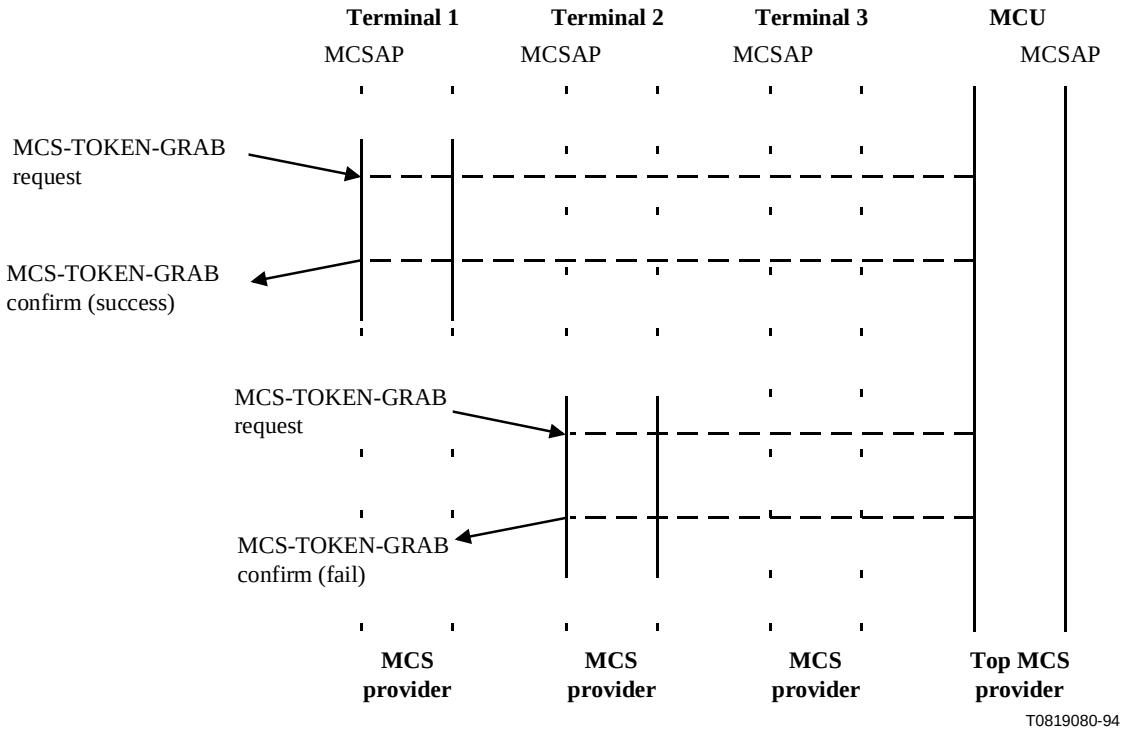


Figure A.7/T.122 – Two users trying to grab the same token

A.3 MCS connection release phase

In the connection release phase, the user leaves the channels it belongs to (MCS-CHANNEL-LEAVE, MCS-CHANNEL-DISBAND primitives), detaches (MCS-DETACH-USER primitive), and disconnect (MCS-DISCONNECT-PROVIDER primitive). If any of the above primitives are not executed, they will be automatically generated when the next in the series of connection release primitives is executed.

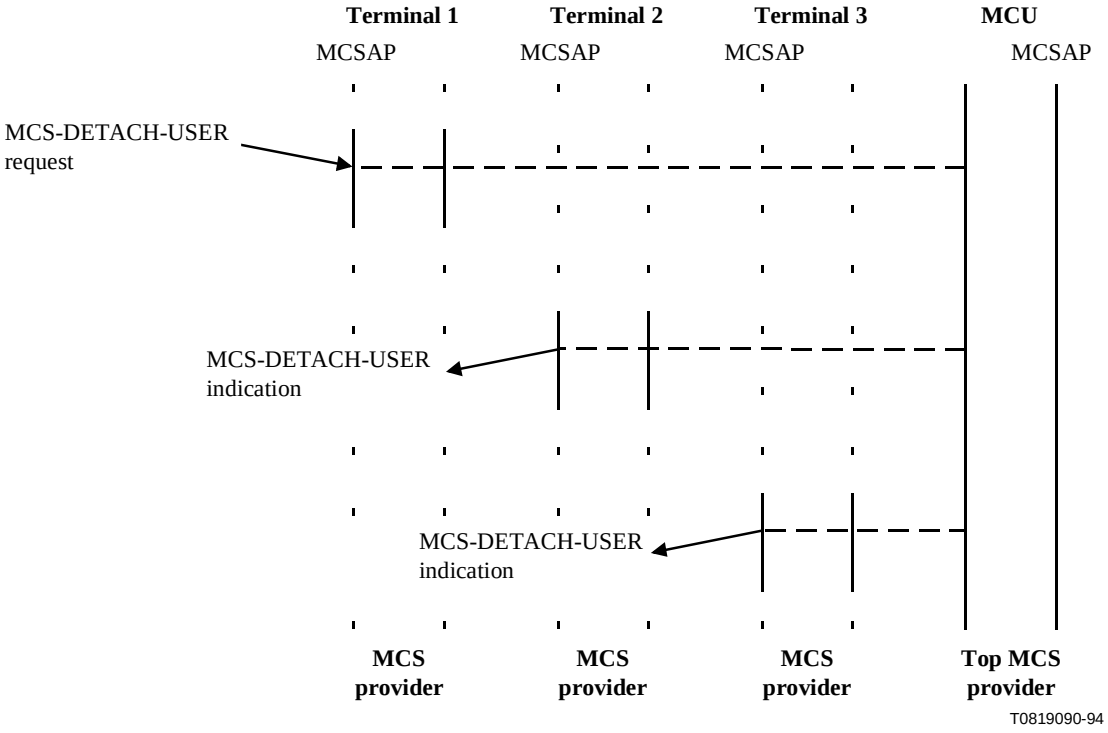


Figure A.8/T.122 – User detaching from a domain

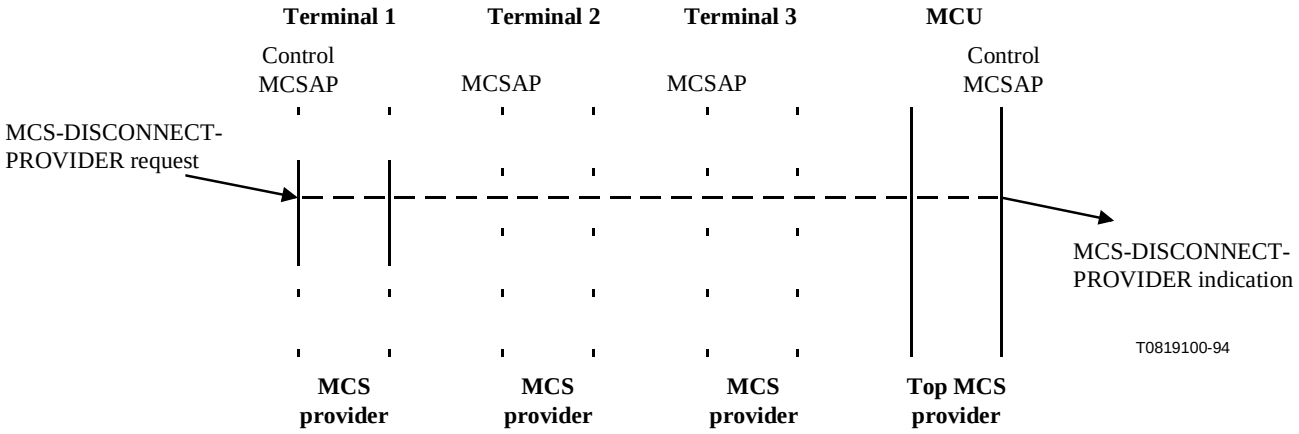


Figure A.9/T.122 – User disconnecting after detaching

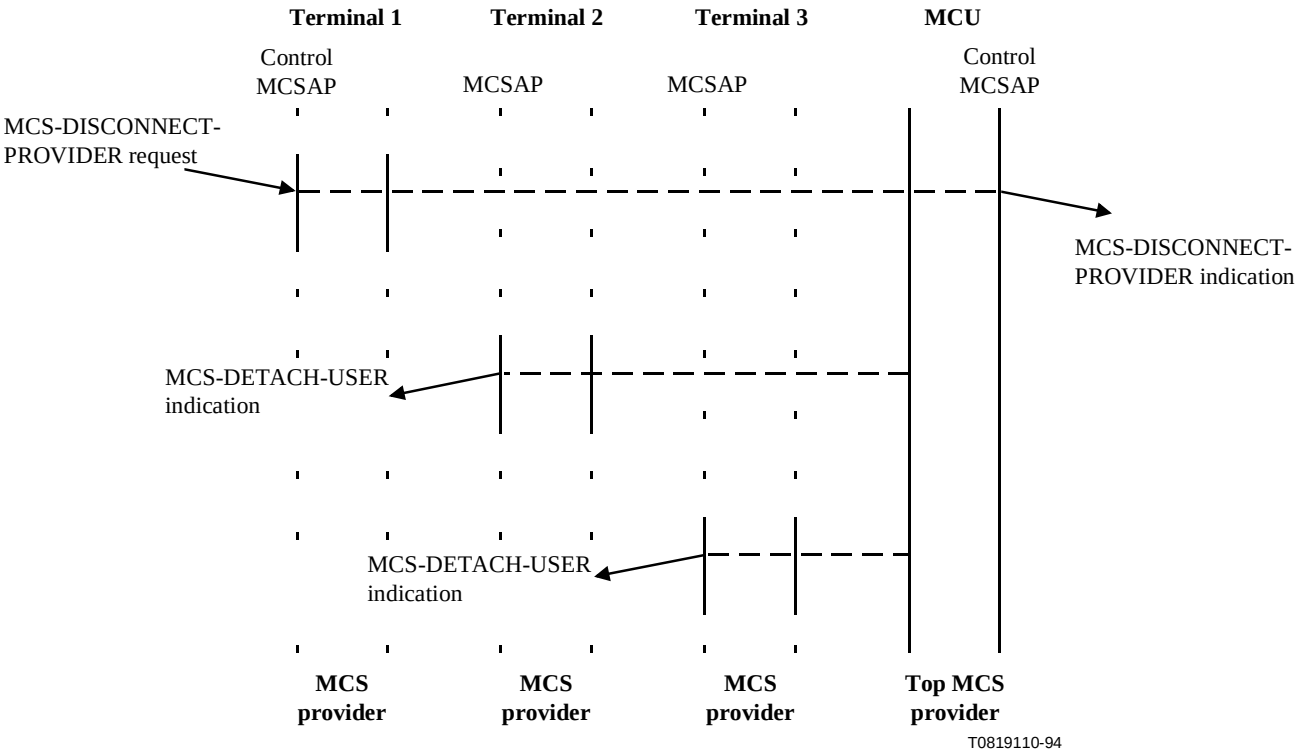


Figure A.10/T.122 – User disconnecting without detaching

APPENDIX I

Distributed token control

The definition of the token services in this Recommendation offers a centralized approach for controlling tokens. The realization of distributed control using the defined services is left for further study.

ITU-T RECOMMENDATIONS SERIES

Series A	Organization of the work of the ITU-T
Series B	Means of expression: definitions, symbols, classification
Series C	General telecommunication statistics
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	TMN and network maintenance: international transmission systems, telephone circuits, telegraphy, facsimile and leased circuits
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Telephone transmission quality, telephone installations, local line networks
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	**Terminals for telematic services**
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks and open system communications
Series Y	Global information infrastructure
Series Z	Programming languages