Video On Demand Session Management

Reuven Cohen
HP Israel Science Center*

and

Yee-Hsiang Chang
Interactive Video Initiative Laboratory†

Keywords: video-on-demand, session protocols, call protocols, interative service, networks

Abstract

A Video-on-Demand (VoD) system provides a service which enables a user of the system to request in real time the transmission of live video from a collection of available video material. A VoD system is probably one of the most complex services to be provided over B-ISDN networks. Such a system consists of several independent service gateways and video servers with which the user Set-Top Box needs to interact during the session lifetime. The paper shows how a VoD session can be modeled as a Finite-State-Machine (FSM). The states in the VoD session FSM represent the calls comprising the session. Every call provides the STB with some preliminary service, or with the selected video application stream. The transitions among states represent call set-up and take-down protocols. The FSM model is an important design tool, which illuminates the various options and trade-offs involved in the design of a VoD session.

*Address: HP Israel Science Center, Technion City, Haifa 32000, Israel. Email: rcohen@hp.technion.ac.il

†Address: Hewlett-Packard Laboratories, 1501 Page Mill Road, Palo Alto, CA 94304-1126, U.S.A. Email: yhc@hpl.hp.com

1 Introduction

Video-on-Demand (VoD) is probably one of the most complex services to be provided over B-ISDN networks. One reason for this is that VoD comprises several distinct services, and each service requires a different network Quality of Service. Another reason is that a VoD system consists of many entities that need to communicate with each other before, during and after the delivery of a VoD application. Among these elements are user Set-Top Boxes (STBs), video-servers, a network bandwidth manager, and service gateways. However, the main reason for the complexity of a VoD system is that users have to navigate through the network in order to select a service provider, and then through the service provider domain in order to select the required video application.

Following the U.S. FCC regulation, a service gateway called Level-1 GateWay (L1GW) or Video Dial Tone Gateway (VDTG), is defined [4, 5]. A L1GW resides inside the network and is part of the network equipment. It is supposed to provide the menuing and selection for the consumer to choose from multiple service providers. The industry also envisions another service gateway, which is the Level-2 GateWay (L2GW) [3]. Every service provider has one or more L2GWs, which resides outside the network as customer premises equipment. The L2GWs are the front-end provided by the service provider for the consumer to select from multiple interactive services.

During the lifetime of a VoD session, the user needs to get several preliminary services before the desired VoD application can be delivered. These services, on one hand, are provided by different entities, but, on the other hand, are strongly dependent on each other. For instance, the user needs to get from the L1GW a white-page menu in order to select a specific service provider. Then, the user needs to get from the L2GW of the selected service provider menuing and selection to choose content.

There are several documented solutions available today [3, 11] for the session protocol. However, those protocols are proprietary and the specifications are still evolving due to many unresolved issues. Currently, the industry is pushing standard protocols in various forums [12, 9, 10].

The purpose of the present paper is two-folds. First, the paper shows how a multiple-call session can be modeled using a Finite-State-Machine (FSM). In case of a Video-on-Demand session, every state in the machine represents a call between the STB and another entity, during which some service is delivered to the STB. The delivered service is either a preliminary service, or the ultimate video application. The transitions among states represent distributed protocols for set-up and take-down of calls.

The second purpose of the paper is to use the multiple-call session model in order to illuminate and discuss the options and trade-offs involved in the design of a VoD session. To this end, we consider a VoD session during which two preliminary services must precede the delivery of a selected video application. Though this paper concentrates upon the provisioning of a VoD service, the suggested session model is applicable for other possible interactive services, as home shopping, video-game on demand, news on demand, etc.

The rest of this paper is organized as follows. Section 2 presents preliminary definitions of key terms. Section 3 shows how a multiple-call session can be represented as a Finite-State-Machine (FSM), with a state for each of the various calls comprising the session. Section 4 and Section 5 address the transitions into two important states of the VoD session. Section 6 deals with two issues related to VoD session management, which can be more easily analyzed using the FSM model: the timing for setting-up the connections needed during a session, and the timing for invoking various authentication and authorization programs. Section 7 concludes the paper.

2 Definitions

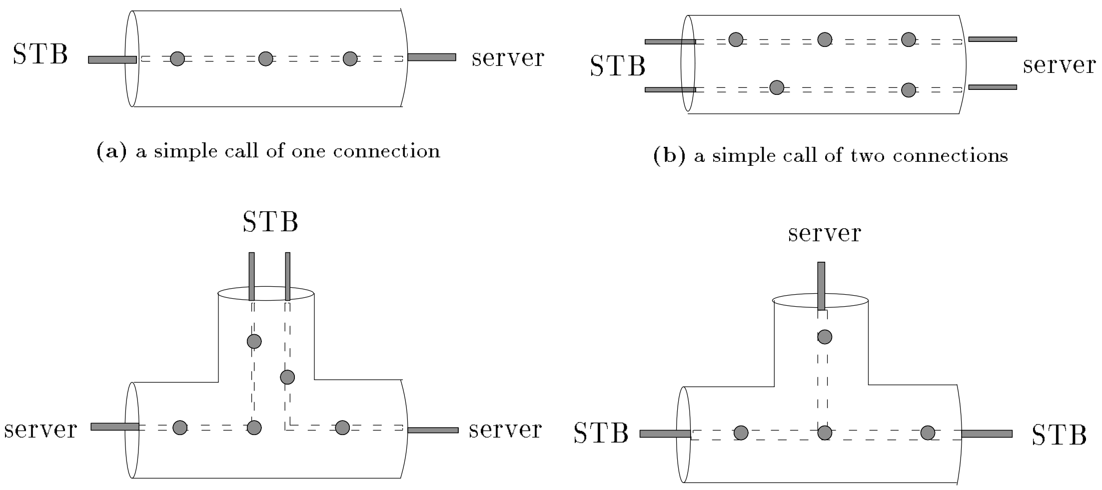
Sessions, Calls and Connections

Suppose that at some time t', a user tunes his STB to the VoD channel. Let t'' be the first time after t' when the STB is tuned to another channel or is turned off. We say that during time period $[t', t'']$ the STB was involved in a single *VoD session.* We represent each session by means of a single contiguous Finite State Machine (FSM). The states of the session FSM represent *calls.* A call is a logical association between the STB and a service provider. It is viewed as an information communication pipe that delivers an instance of a service to the user. A call is established by means of a distributed control procedure, referred to as call set-up protocol. It is canceled by means of another distributed control procedure, referred to as call take-down protocol.

Though the number of states (representing calls) in the session FSM is finite, the number of calls a STB may actually participate during the session lifetime is not necessarily bounded, because some states (calls) can be entered more than once during a single session, depending on the specific session specifications. On the other hand, in some irregular cases, the session may have *no* call. This might happen, for instance, if the first entity with which the STB should have a call during the session is down, in which case the first service cannot be delivered, and all the other services depend on the first one.

A *connection* is an end-to-end bit pipe, associated with a set of Quality of Service (QoS) parameters, over which call information is exchanged among the call end entities. In ATM networks, for instance a connection (referred to as Virtual Channel) is a switched path formed by the hardware of lower layer protocols. In a TCP/IP network, on the contrary, a connection (referred to as TCP connection) is formed by the software of upper layer protocols. In order to realize a call, one or more connections are needed. A telephone call usually needs a single connection across the network, but a more complex call, like digitized video, may need more connections: one for video information, another one for audio information, and a third one for control information. Figure 1(a) shows a simple (two-way) call consisting of one connection. In a TCP/IP network, the bullets along the connection represent IP gateways [7], whereas in an ATM network they represent VC switches [8]. Figure 1(b) shows a simple call consisting of two connections. Some of the calls can be multiple-point calls, where more than two entities are involved. Since this paper considers interactive service networks, a three-way call may consist of one STB connected to two servers (Figure 1(c)) or two STBs connected to one server by means of multicast connections (Figure 1(d)). In the latter case, however, each STB is associated with a different session, and the three-way call can be considered as two simple (two-way) calls. Thus, the 1 to 1 relationship between calls and STBs is retained.

One may observe here a hierarchy of sessions over calls over connections, with a 1 to N relationship between sessions and calls (i.e. one session consists of N calls) and a 1 to N relationship between calls and connections. Since we assume that all the services, preliminary



(a) a simple call of one connection

(b) a simple call of two connections

(c) a multiple-point call with two servers

(d) a multiple-point call with two users

Figure 1: Call Examples

as well as ultimate, provided during a single session are destined for a single STB, the STB is an end-point of every call. Thus, only one call can be active at a time, though more calls can be non-active but *alive.* A call is said to be *active* when its connections transfer data. A call is said to be *alive* if it has not been taken down since the time it was set up.

Figure 2 depicts two sessions. The session in Figure 2(a) is a relatively simple one. At the beginning of the session call-1 is set up[1] in order to deliver some service to the STB. Then, call-1 is taken down and another call, referred to as call-2, is set up. Sometime later, call-2 is taken down, and a third call, call-3, is set up. When call-3 is taken down, the entire session comes to an end. The session in Figure 2(b) is more complex. At the beginning of the session, call-1 is set up. However, before call-1 is taken down, call-2 is set-up . Thus, there are two alive calls, though only one of them can be active. Then, both calls are taken down and call-3 is set-up. After call-3 is taken down, another instance of call-1 is set-up. Finally, call-1 is taken down and the entire session comes to an end.

Dependencies of Calls

Every two calls performed during a single session are either *dependent* or *independent* of each other. We say that a call, call-2 say, depends on another call, call-1, if call-2 can be active only after call-1 is active during the lifetime of the same session.

As an example, consider a user calling a telephone service provider. For the sake of this example, suppose that the user has to pass some authentication and authorization process. This process, can be considered as a call which delivers the authentication and authorization service to the user. During this call, the user is asked some questions (like "please enter

[1]Note that "1" does not represent a sequence number of the call during a session, but an identity of the call.

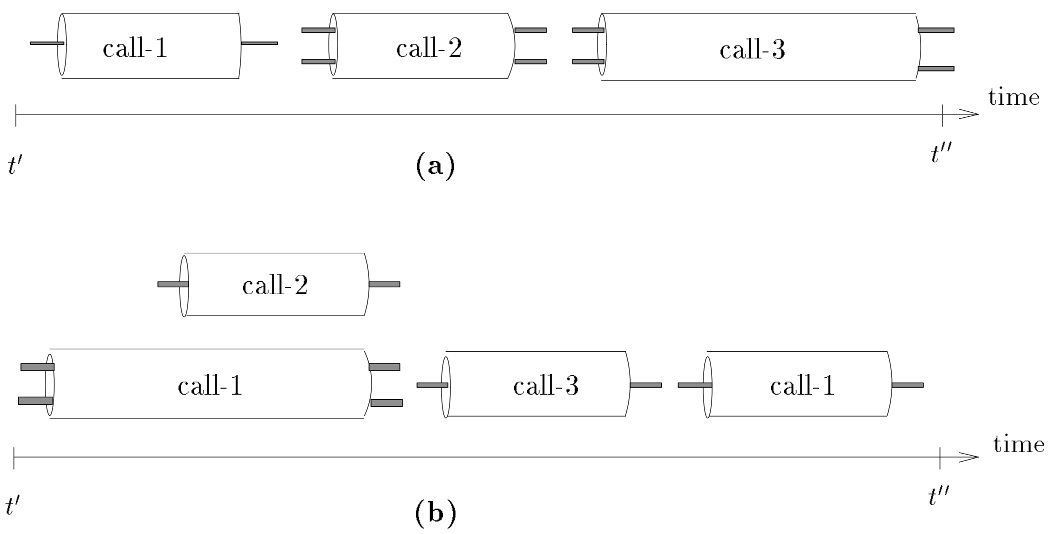


Figure 2: Session Examples

your 8-digit identification code") and needs to provide answers. Once this call is successfully completed, the user is authorized to conduct $N \geq 0$ telephone calls. We consider the interval between the time the user starts the authentication and authorization process and the time he finishes the last telephone call as a single session. This session consists of $N + 1$ calls: an authentication and authorization call (call-1) followed by N telephone calls (N instances of call-2). In this example, call-2 depends on call-1 because a telephone call cannot be conducted before the authentication and authorization call is completed.

Dependencies of Entities

Consider a call, call-i say, for which entity E is an end point. Then, for every entity $E' \neq E$:

- E is said to be *call-i independent* of E' if E' is not involved in the process of admitting call-i.
- E is said to be *call-i semi-dependent* of E' if both E and E' are involved in the process of admitting call-i.
- E is said to be *call-i dependent* of E' if E' is involved and E is not involved in the process of admitting call-i.

The admission of a call is performed by means of a call set-up protocol, executed among some entities in the network. Every entity in the network, including the call end-points, may participate or not participate in the call set-up protocol. Entity E is call-i independent of any entity E' which does not participate in the set-up protocol of call-i. If E' participates in the set-up of call-i then each of the three cases may apply, depending on the role of E'

in the set-up protocol and on whether or not entity E also participates in the protocol. In particular, there might be a case where E' participates in the set-up protocol but E is call-i independent of E' because E' is unable to affect the call admission decision. Nevertheless, in the following we shall assume that:

- E is said to be call-i independent of E' if E participates in call-i set-up protocol and E' does not participate.
- E is said to be call-i semi-dependent of E' if both E and E' participate in call-i set-up protocol.
- E is said to be call-i dependent of E' if E does not participate in call-i set-up protocol and E' participates.

If both end points of call-i are *call-i dependent* of other entities, then third party connection signaling protocol must be employed. This protocol is needed when a network connection between two end terminals is initiated not by an end terminal. The third party can be a node inside the network, in which case the third party connection protocol is either a proprietary protocol used by the network operator or a standardized NNI (Network-to-Network Interface) protocol. The third party can also be a node outside the network, in which case the third party connection signaling protocol has to be part of the standard UNI (User-Network Interface). The ATM Forum is currently standardizing the latter case.

3 A Multiple-Call Session Model

This section shows how a multiple-call session can be represented as a Finite-State-Machine (FSM). The FSM describes the various calls comprising the session and the transitions among them. To present our model, which can be adopted for every multiple-call session, we assume that two preliminary services should be provided to the user before a video application is selected.

Consider the following VoD session *example.* After a user tunes his STB to the VoD channel, he may quit or ask to view the list of service providers (white-page menu). In the latter case, a call is established between the STB and the L1GW. This call, during which the white-page menu is delivered to the user STB, will be referred to as call-1. The user may now select a service provider, in which case call-1 is taken down and another type of call, referred to as call-2, is established between the STB and the L2GW associated with the selected service provider. During call-2, the L2GW provides the user with a selection of available video application (yellow-page menu), to which preview information can be attached. The user may now quit or select some video application. In the latter case a call is established between the user STB and the video-server selected by the L2GW to deliver the requested application. This call is referred to as call-3.

The VoD session described so far is represented by the FSM depicted in Figure 3. Such an FSM is associated with every STB session. The session FSM has 5 states: OFF, NO-SERVICE, SERVICE-1, SERVICE-2 and SERVICE-3. The first state, OFF, indicates that the last session has already ended and a new one has not been initiated yet. The NO-SERVICE state is entered when the STB is tuned to the VoD channel, and the session starts. SERVICE-1 is the state of the session when call-1 is active. SERVICE-2 is the state

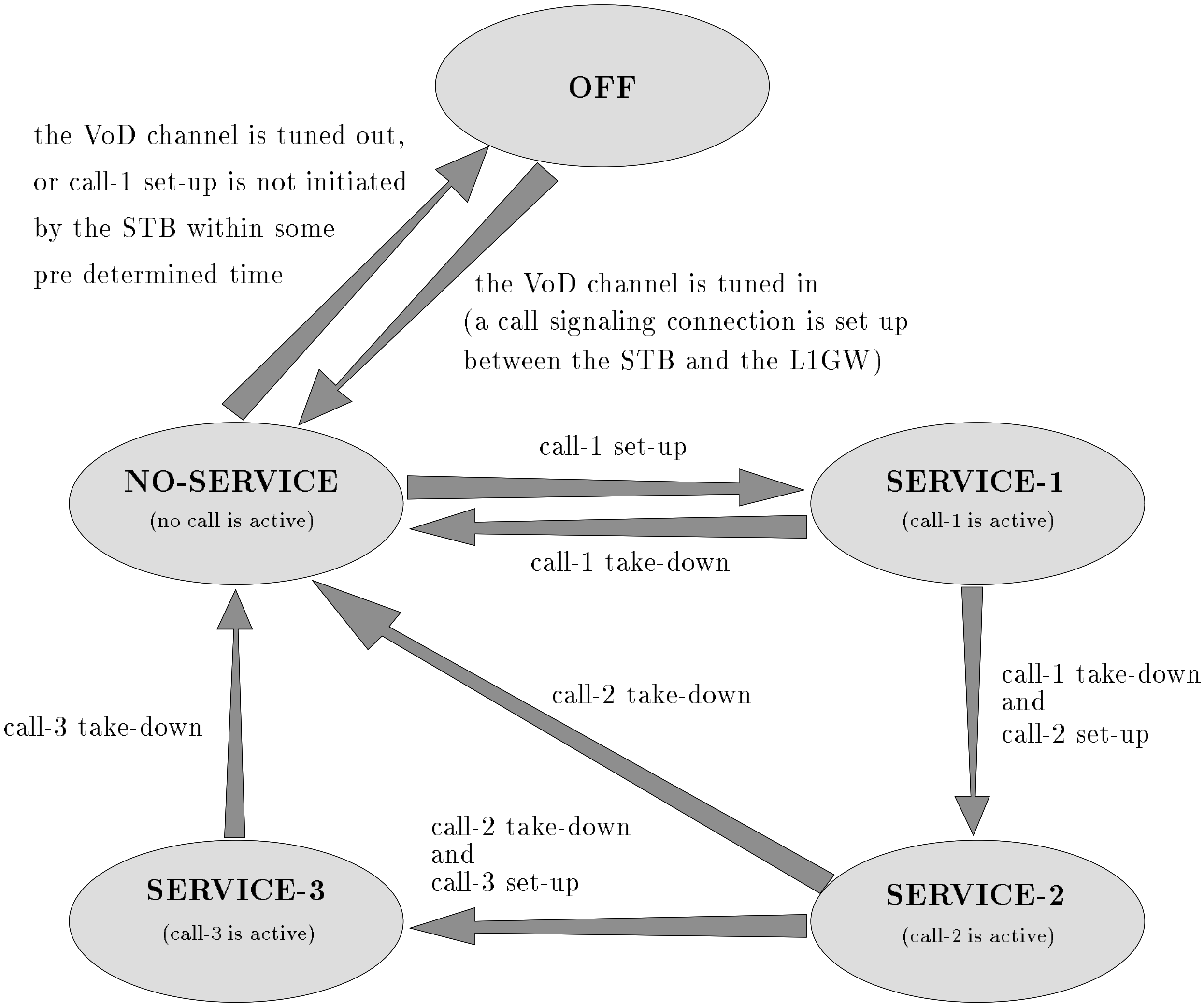


Figure 3: A Finite-State-Machine Example for a Video-on-Demand Session

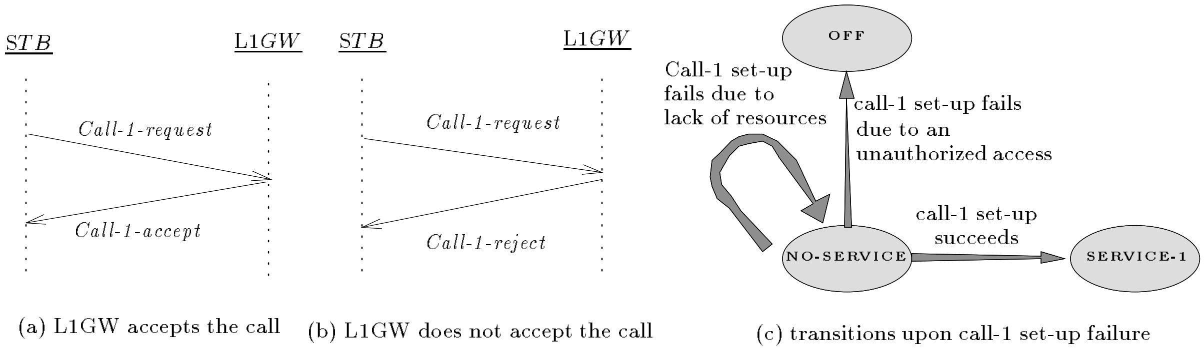


Figure 4: Call-1 Set-Up Protocol

of the session when call-2 is active. Finally, SERVICE-3 is the state of the session when call-3 is active.

Note that the FSM shown in Figure 3 is incomplete, since only key transitions are shown. In particular, the transitions labeled as "call set-up" reflect the case where the set-up is successful, but not the case where it fails due to lack of network resources, unauthorized access, or any other reason. As explained later, when a call set-up fails, the session may proceed in several ways. Note also that the FSM in Figure 3 is only an example for a possible session model. Another session model may decide not to take down call-1 upon entering SERVICE-2 from SERVICE-1. Yet another model may not allow a direct transition from SERVICE-1 into SERVICE-2, but only an indirect transition through the NO-SERVICE state. All these are important design issues, to be addressed later.

As shown in Figure 3, the transitions between the various states usually represent call set-up and take-down protocols. The purpose of a call set-up protocol is to request the provision of a specific service and to allocate the resources needed by that service. The purpose of a call take-down protocol is to stop provisioning the service and to release the allocated resources. The entities participating in a call set-up or call take-down protocol are determined according to the resources needed by the call, which depend on the call type.

Consider the set-up protocol for call-1. During call-1, the L1GW connected to the STB is supposed to deliver the white-page menu to the STB. This is a basic service, which does not require much of the network resources. Assuming that such a service can be locally approved by the L1GW, the set-up protocol is essentially a simple two-way handshake protocol as depicted in Figure 4. Figure 4(a) shows the case where the L1GW accepts the call, in which case the transition in Figure 3 from NO-SERVICE into SERVICE-1 is enabled. Figure 4(b) shows the case where the L1GW does not accept the call for some reason. The transition made in such a case is not shown in Figure 3 because there are two reasonable options: staying in NO-SERVICE or returning to OFF. The decision can be made according to a parameter returned by the L1GW in the *Call-1-reject* message, indicating the failure cause (Figure 4(c)).

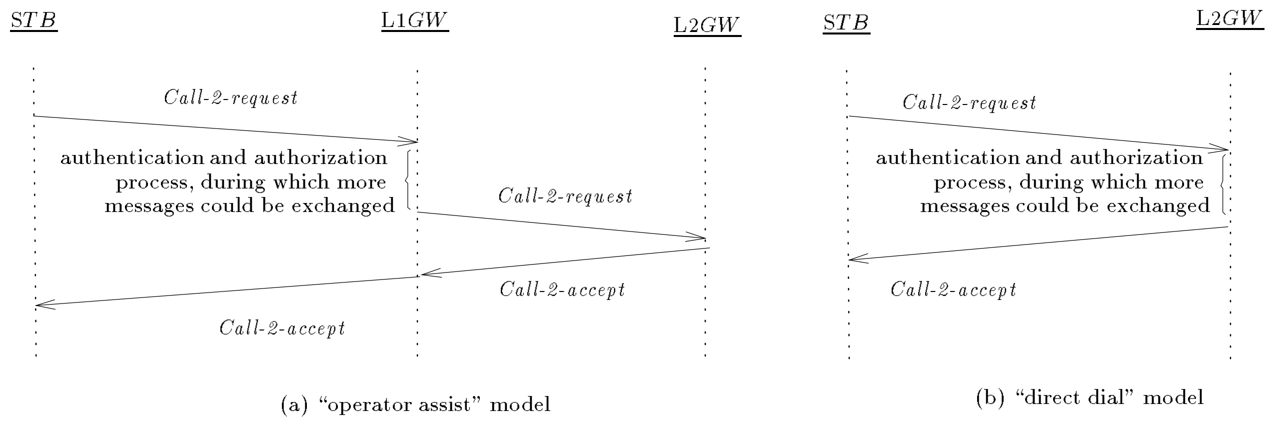


Figure 5: Operator Assist Vs. Direct Dial

4 The Transition into SERVICE-2 State

The transition of the STB FSM into SERVICE-2 state depends, among other things, on the relationship between the L1GW and the L2GW providing the service during call-2. Since the L1GW is owned and managed by the network operator, whereas the L2GW is envisioned as customer premises equipment, we do not consider a case where the L2GW is call-2 dependent of the L1GW. This leaves the following two possibilities:

- The L2GW is call-2 independent of the L1GW.
- The L2GW is call-2 semi-dependent of the L1GW.

The first possibility is more likely when the L2GW has its own mechanisms for authentication, authorization and billing. When this is not the case, these functionalities are provided by the L1GW on behalf of the L2GW. To this end, the L1GW must be involved in the transition of the STB into SERVICE-2 state.

If the L2GW is call-2 independent of the L1GW, the set-up of call-2 is performed by means of "direct dial" mode [6]. Direct dial is similar to today's directory service (e.g. 411 call): the user gets the destination number from a directory server, and then dials by himself. In contrast, if the L2GW is call-2 semi-dependent of the L1GW, the set-up of call-2 is performed by means of "operator-assist" mode [6]. In this model, the user first connects to the operator and tells him the destination number. The call is then made by the operator on behalf of the user. In the context of VoD, the role of the operator is played by the L1GW.

A conceptual representation of the two models is given in Figure 5. In the operator assist model (Figure 5(a)), the STB sends a *Call-2-request* message to its L1GW. One of the parameters attached to this message is an explicit or implicit address of the L2GW to which the STB wishes to be connected. This address has been determined by the STB using the white-page menu provided by the L1GW while the STB was in SERVICE-1. When the L1GW receives a *Call-2-request* message, it may invoke some authentication and authorization procedure, to make sure that this call request should be forwarded to the L2GW. This procedure may dictate the exchange of more messages between the STB and the L1GW (not shown in Figure 5(a)). If the L1GW approves the request, the call request message is forwarded, with some possible modifications, to the appropriate L2GW. Assuming that the L2GW agrees to admit the call, a *Call-2-accept* message is sent by the L2GW to the L1GW and forwarded to the STB. Upon receiving this message, the STB enters SERVICE-2 state. In the direct dial model (Figure 5(b)), the STB sends the *Call-2-request* message directly to the L2GW, and the latter responds with *Call-2-accept* if the call can be admitted, or with *Call-2-reject* otherwise.

The connections needed between the STB and the L2GW for call-2 are not necessarily affected by the call set-up model. In particular, even if the operator assist model is used, these connections do not have to traverse the L1GW.

Figure 6 shows the various options effective (a) when the L2GW is call-2 independent of the L1GW and (b) when the L2GW is call-2 semi-dependent of the L1GW. As the figure shows, in both cases the transition from SERVICE-1 state into SERVICE-2 state can be made directly, or indirectly through NO-SERVICE state. Figure 7 depicts both options. Note that a direct transition (Figure 7(a)) is the option employed in Figure 3. The decision between direct transition and indirect transition depends upon three factors as follows:

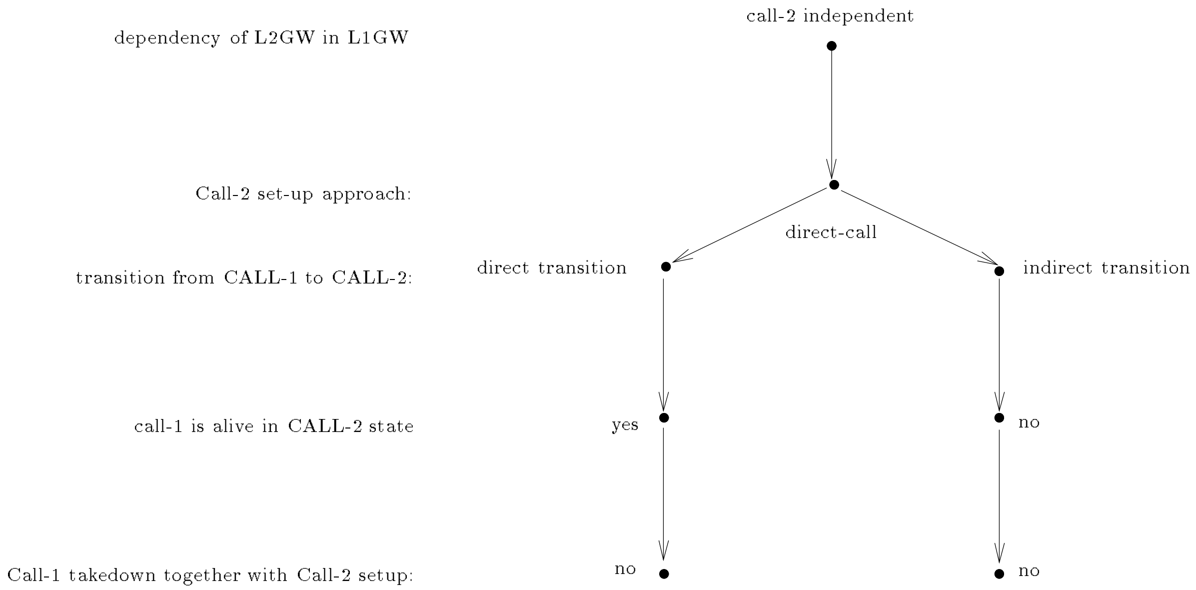
1. Whether the L2GW is call-2 independent of the L1GW or it is call-2 semi-dependent of the L1GW.

2. Whether the STB is intelligent (i.e. has the rendering for display, database for query and computing power for navigation) or not[2].

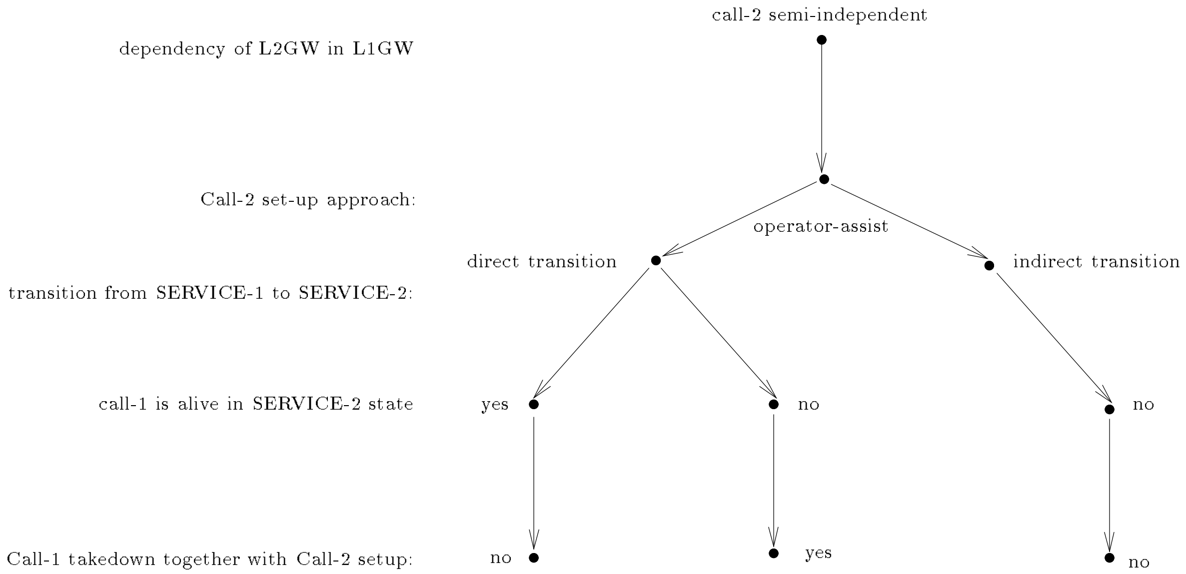
3. Whether call-1 should remain alive in SERVICE-2 state, or it should be taken down during the transition into SERVICE-2. The two options are depicted in Figure 8. Consider the FSMs of Figure 8(a) and (b) first. In both cases SERVICE-2 state is entered only after call-1 take-down protocol is performed. The take-down protocol can be as simple as the two-way handshake set-up protocol depicted in Figure 4: a *Call-1-cancel* message is sent by the STB to the L1GW, and the latter responds with *Call-1-cancel* of its own. If the user wishes to return from SERVICE-2 into SERVICE-1 (e.g., since he is not satisfied with the selection offered by the L2GW of the selected service provider, and, therefore, wishes to choose another service provider), call-2 take-down protocol and call-1 set-up protocols must be invoked. The simplest way to achieve this is to have the STB transit from SERVICE-2 state back to NO-SERVICE state, by executing call-2 take-down protocol, and then from NO-SERVICE into SERVICE-1, by executing call-1 set-up protocol.

   In the FSM of Figure 8(c), in contrast, call-1 is not taken down when the session enters SERVICE-2. Consequently, a direct transition from SERVICE-2 into SERVICE-1 is possible by invoking call-2 take-down protocol. The main advantage of this approach is that it reduces the time and overhead for re-entering SERVICE-1. The main disadvantage is that network resources allocated to call-1 cannot be released while they are

[2] Though the STB intelligence is a factor that affects the design of the VoD session, it will probably not mandate which choice to take. However, we assume the design takes the STB intelligence into consideration.

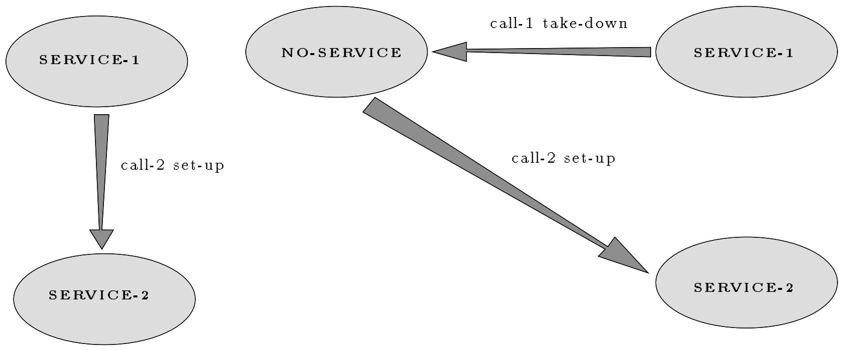


(a) when the L2GW is call-2 independent of the L1GW



(b) when the L2GW is call-2 semi-independent of the L1GW

Figure 6: The Various Options for the Transition into SERVICE-2



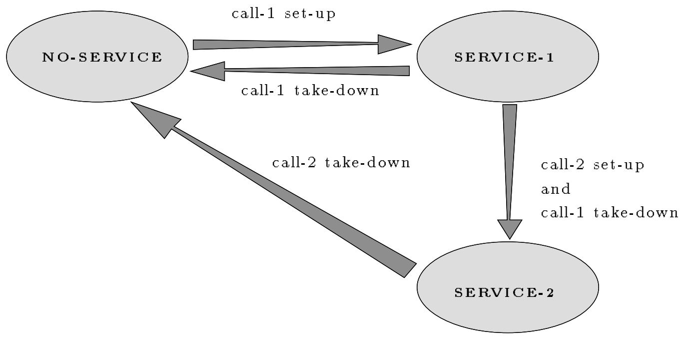
(a) direct transition (b) indirect transition

Figure 7: Direct Vs. Indirect Transition from SERVICE-1 to SERVICE-2

not used in SERVICE-2 state. A design decision can be made after user behavior is analyzed. If the users do not tend to return from SERVICE-2 into SERVICE-1 very often, and if the time period during which they stay in SERVICE-2 before returning to SERVICE-1 is relatively long, call-1 should be taken down upon entering SERVICE-2. If, however, the users return very often from SERVICE-2 into SERVICE-1, after staying a short period of time in SERVICE-2, keeping call-1 alive in SERVICE-2 might yield better performance.

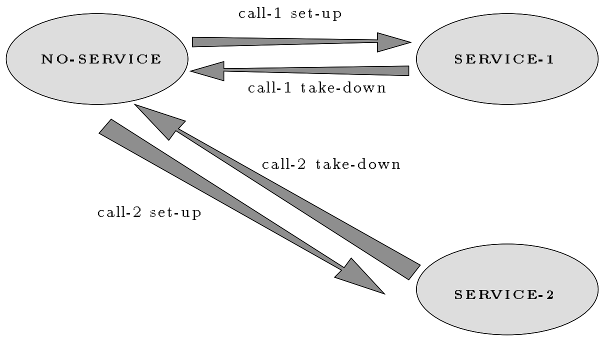
The eight options induced by factors 1-3 are discussed in the following. Consider first the cases where the L2GW is call-2 independent of the L1GW. Recall that in these cases, the L1GW does not participate in call-2 set-up protocol. If call-1 has to be kept alive in SERVICE-2, then regardless of the STB intelligence the STB stays in SERVICE-1 until selecting a service provider. Then, by invoking an instance of call-2 set-up protocol (as described in Figure 5(b)), the STB moves directly into SERVICE-2 state. The relevant part of the FSM are shown in Figure 8(c).

If, however, call-1 should not be kept alive in SERVICE-2 state, an indirect transition from SERVICE-1 to SERVICE-2 via NO-SERVICE should be used (Figure 8(b)). This is clear for the case where the STB is intelligent, since in such a case it can move from SERVICE-1 back to NO-SERVICE immediately after getting the white-page information from the L1GW. The take-down of call-1 is performed by the L1GW and the STB. Then, upon selecting service provider, the STB can move into SERVICE-2 state using the set-up protocol described in Figure 5(b). For the case where the STB is not intelligent, the transition into NO-SERVICE state is necessary in order to avoid another state, as explained in the following. When the STB selects a service provider, call-1 has to be taken down and call-2 has to be set-up. These two protocols cannot be merged in this case because the L1GW is involved in the take-down of call-1 but not in the set-up of call-2 (since the L2GW is call-2 independent of the L1GW). Thus, call-1 take-down has to be invoked first, and then call-2 set-up can be invoked. Since a transition from SERVICE-1 into NO-SERVICE, triggered by call-1 take-down, must always exist, a transition from NO-SERVICE into SERVICE-2 triggered by call-2 set-up would be sufficient, as Figure 8(b) shows.



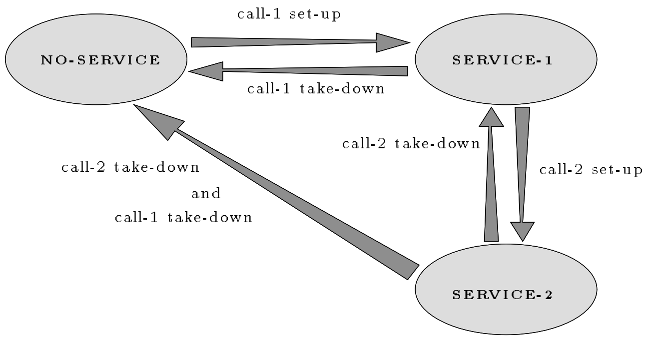
(a) call-1 is not kept alive in SERVICE-2

(direct transition from SERVICE-1 to SERVICE-2)



(b) call-1 is not kept alive in SERVICE-2

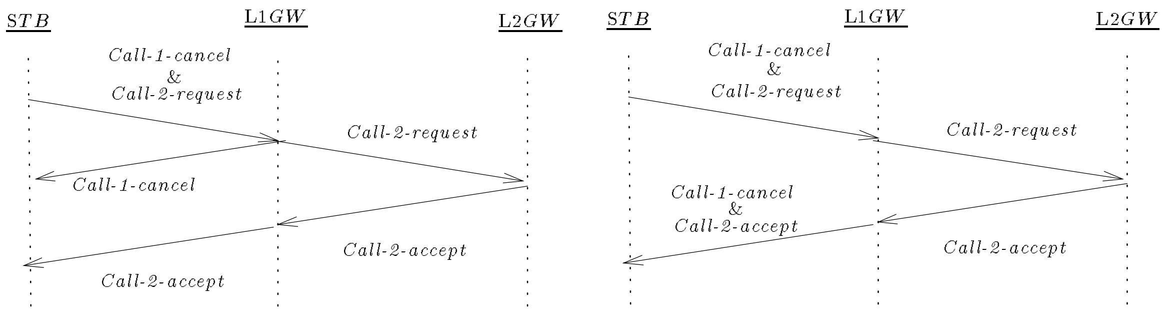
(indirect transition from SERVICE-1 to SERVICE-2)



(c) call-1 is kept alive in SERVICE-2

Figure 8: Moving from SERVICE-2 to SERVICE-1 and Vice Versa

Next, consider the cases where the L1GW is call-2 semi-dependent of the L2GW. If call-1 has to be kept alive in SERVICE-2 state, then, regardless of the STB intelligence, a direct transition is made from SERVICE-1 into SERVICE-2 (Figure 8(c)). If call-1 should be taken down upon entering SERVICE-2 state, the transition from SERVICE-1 into SERVICE-2 depends on the STB intelligence. If the STB is intelligent, it leaves SERVICE-1 and returns to NO-SERVICE as soon as it gets the white-page menu from the L1GW. Then, when the STB selects a service provider, the call-2 set-up protocol shown in Figure 5(a) is invoked and the STB enters SERVICE-2 (Figure 8(b)). If the STB is not intelligent, i.e. has no capability for local rendering, database query, and navigation, it needs to stay in SERVICE-1 until selecting a service provider. Then, call-1 should be taken down and call-2 be set-up (Figure 8(a)). Since the L2GW must participate in both protocols, they can be integrated into a single one which causes a direct transition from SERVICE-1 into SERVICE-2. Two conceptual implementations of the integrand protocol are depicted in Figure 9. An interesting case is when call-2 cannot be admitted. If the option depicted in Figure 9(b) is employed, then when the L1GW receives *Call-2-reject* message from the L2GW, it can determine



(a) semi-integration

(b) full integration

Figure 9: Combining Call-2 Set-Up With Call-1 Take-down

whether to take call-1 down, or to keep it alive. The decision can be made according to a parameter attached by the STB to the *Call-2-request* message sent to the L2GW. In such a case, depending on the STB request and on the response of the L2GW, the STB may (a) enter SERVICE-2 state; (b) remain in SERVICE-1 state; or (c) return to NO-SERVICE state. If the option depicted in Figure 9(a) is employed, call-1 cannot be kept alive after call-2 is rejected.

5 The Transition into SERVICE-3 State

The transition into SERVICE-3 is more complicated than the transition into SERVICE-2. One reason is that the service provided during call-3 potentially demands more network resources, during a much longer period of time than the service provided during call-2. Thus, other entities need to specifically identified in the set-up of call-3. In particular, the network bandwidth manager, to which nodes need to apply in order to get high-speed channels for a long duration, will have to participate in the set-up of call-3.

Unlike the providers of call-1 and call-2 (L1GW and L2GW, respectively), the video-server does not necessarily need to participate in the set-up of the call it provides (call-3). This is because most of the activities associated with call set-up, like billing, authentication and authorization, can be provided to the video-server by its L2GW.

In order to reduce the number of options needed to be explored, the following assumptions are made:

1. The L2GW must participate in call-3 set-up. We make this assumption because we view the L2GW as the center of the service provider complex, where most of the management and control activities are performed. An important implication of this assumption is that, regardless of the STB intelligence, the transition into SERVICE-3 state will be made *only* from SERVICE-2 state. Another implication is that the video-server does not necessarily have to participate in call-3 set-up.

2. A network bandwidth manager must participate in call-3 set-up protocol[3].

3. Call-2 is not kept active upon entering SERVICE-3 state. The reason for this is that call-3 is envisioned as a relatively long call, and the probability that the STB will want to quit in an early stage of this call is considered as relatively low, especially if a preview service is provided in SERVICE-2 state.

Examples of call-3 set-up protocol are presented in Figure 10. In parts (a) and (b) it is assumed that the video-server is call-3 semi-dependent of its L2GW, whereas in part (c) it is assumed that the video-server is call-3 dependent of its L2GW. The difference between (a) and (b) is that in (a) the L2GW is considered as the pivot of the set-up protocol, whereas in (b) the set-up is more distributed. In (a) the STB sends the L2GW a call-3 request message. The L2GW uses a two-way handshake with the video-server in order to verify that the latter can provide the requested service. After the video-server confirms, the L2GW asks the bandwidth manager to allocate network resources for the call. When the resources are allocated, the L2GW informs the two end-points of call-3, the video-server and the STB, that the call is accepted.

In Figure 10(b), the request for bandwidth allocation is sent to the bandwidth manager by the video-server, rather than by the L2GW. This example, which reduces the set-up complexity, assumes more intelligence in the video-server side. The protocol shown in Figure 10(c) can be acquired by removing the video-server and its associated arrows from either (a) or (b).

From our assumptions that call-3 set-up can be invoked only when the STB FSM is in SERVICE-2 state, and that call-2 is not kept active while SERVICE-3 state is entered, follows that call-2 take-down protocol should be performed along with call-3 set-up. Moreover, if call-1 was kept alive in SERVICE-2 state, this call should also be taken down during the transition into SERVICE-3. Thus:

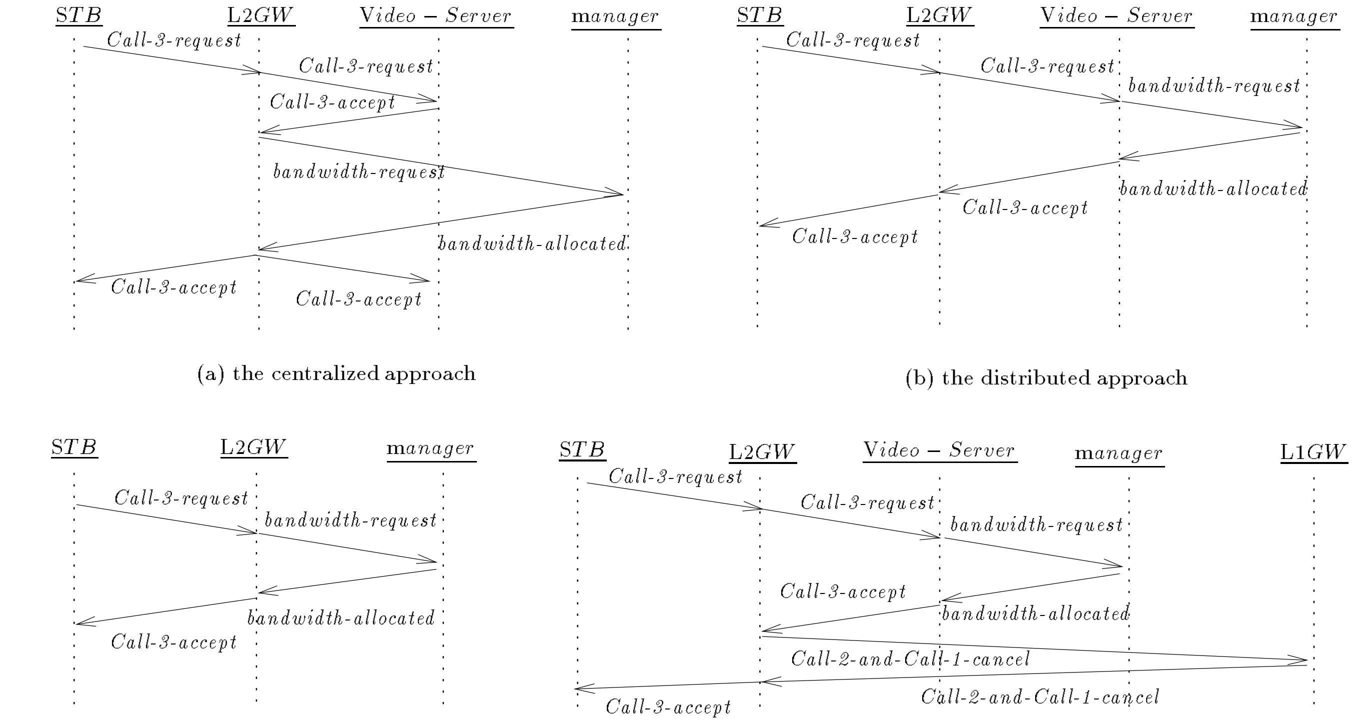
1. If call-1 is kept alive in SERVICE-2 state, then regardless of the dependency of the L2GW in the L1GW: (*i*) both call-1 and call-2 need to be taken down during the set-up of call-3; and (*ii*) the L1GW has to participate in the set-up of call-3. The protocol presented in Figure 10(d) is a modification of the protocol presented in Figure 10(b) with an additional two-way handshake between the L2GW and the L1GW for taking call-1 down.

2. If call-1 is not kept alive in SERVICE-2 state and the L2GW is call-2 independent of the L1GW, then: (*i*) only call-2 needs to be taken down during the set-up of call-3; and (*ii*) the L1GW does not have to participate in the set-up of call-3.

3. If call-1 is not kept alive in SERVICE-2 state and the L2GW is call-2 dependent of the L1GW, then: (*i*) only call-2 needs to be taken down during the set-up of call-3; and (*ii*) the L1GW must participate in the set-up of call-3. In this case, the protocol presented in Figure 10(d) with small changes (i.e. using *Call-2-cancel* messages instead of *Call-2-and-Call-1-cancel* messages) can be employed.

Another issue related to the set up of call-3 is the state the FSM enters upon set-up failure. Since call-2 is taken-down only after call-3 is set up, the best option is staying in SERVICE-2

[3] Though a bandwidth manager was not mentioned as an entity participating in the set-up of call-1 and call-2, we do not exclude this possibility.



(a) the centralized approach

(b) the distributed approach

(c) the video-server is call-3 dependent of the L2GW

(d) call-1 is active while SERVICE-2 is entered

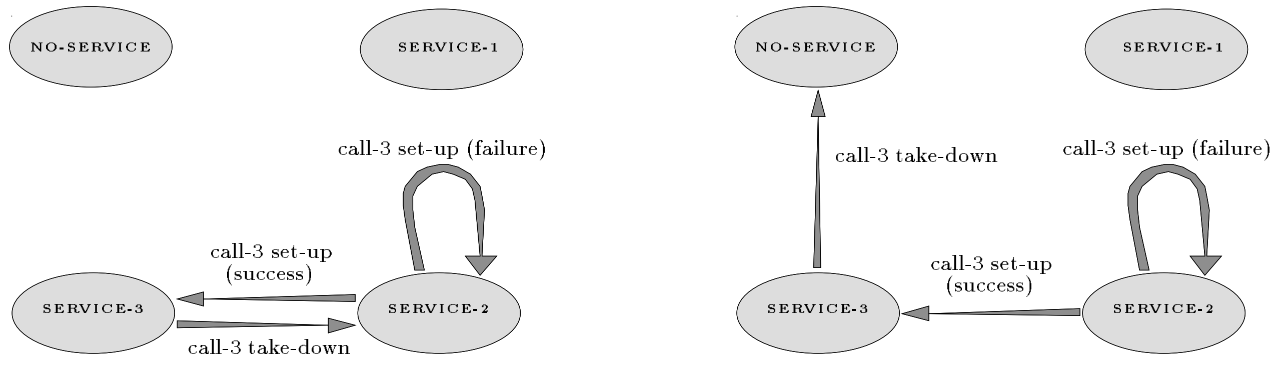
Figure 10: Examples of Call-3 Set-Up

state. This gives the STB an opportunity to select another service or to quit by invoking call-2 take-down protocol.

If the set-up of call-3 succeeds and the STB enters SERVICE-3 state, a transition out of this state, when call-3 ends, can be made into SERVICE-2 state or into NO-SERVICE state. A direct transition from SERVICE-3 back to SERVICE-2 (Figure 11(a)) is possible only in those session models where call-1 is not kept alive in SERVICE-2 state. When this is not the case, a transition from SERVICE-3 must be performed into NO-SERVICE state (Figure 11(b)), and SERVICE-2 state can be entered only after SERVICE-1 is entered.

6 Related Issues: Connection Set-Up and Authorization

This section deals with two issues related to VoD session management, which can be more easily analyzed using the FSM model: the timing for setting-up the connections needed during a session, and the timing for invoking various authentication and authorization programs.



(a) if call-1 is not active in SERVICE-2

(b) if call-1 is active in SERVICE-2

Figure 11: The Transitions Associated With SERVICE-3

The Set-Up of Service and Control Connections

The information exchanges during each call in a switched ATM/STM network require one or more connections. For instance, to provide SERVICE-3 during call-3 state, connections should be established from the video-server to the STB. In some cases, one connection, that carries all types of information, is sufficient. In other cases, where different types of data (voice, video, timing etc.) require connections with distinct Quality of Service parameters, more connections from the video-server to the STB are needed. Similarly, to allow the STB request special operations (like pause or fast forward) to the video-server, a service control connection in the reverse direction is needed as well.

Connections are established in the network by means of connection signaling protocols. Usually, different protocols are defined for the User Network Interface (e.g. Q.930/Q.931 for PSTN, and Q.2931 for ATM [2]) and for the Network Network Interface (e.g. B-ISDN User Part and SS7 ISDN User Part [1]). Using these protocols, connection set-up messages are sent in a tandem along the route over which the connection is to be established, and the routing tables of the switches are updated to enable the switched connection [8].

Since in switched ATM/STM networks a connection is needed before any message is sent, connections should be established not only before the delivery of call information in the various service states (SERVICE-1, SERVICE-2 and SERVICE-3), but also before the exchange of call control information during the transitions among the various states (unless a connection-less service is available). The connections needed in the various SERVICE states will be referred to as *service connections,* whereas those needed for the exchange of control (set-up/take-down) information will be referred to as *control connections.* For instance, in SERVICE-3 state some service connections are needed between the STB and the video-server. However, to enter SERVICE-3 state by executing the set-up protocol depicted in Figure 10(a) as an example, *control connections* are needed between the STB and the L2GW, between the video-server and the L2GW, and between the bandwidth manager and

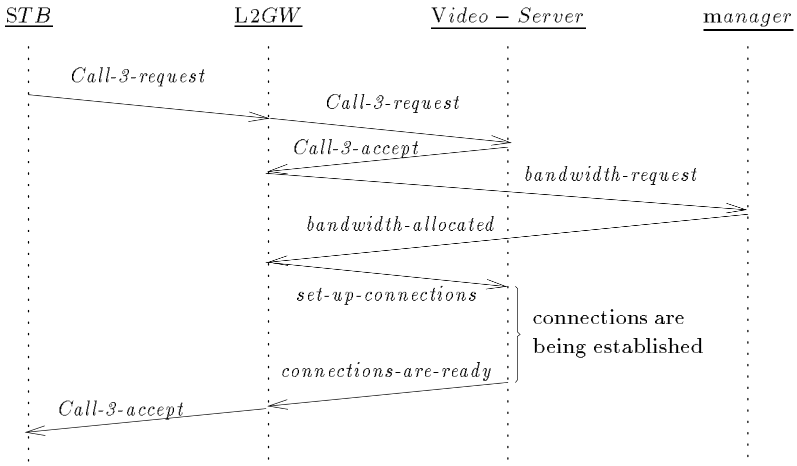


Figure 12: SERVICE-3 Application Connections are Established During Call-3 Set-Up

the L2GW. An important VoD session management decision is when to set up all these connections.

The service connections can be established either after or during the execution of the call set-up protocol. For instance, Figure 10(a) has shown the case where the connections in SERVICE-3 state are established after call-3 set-up protocol is completed; namely, after the STB and the video-server receive a *Call-3-accept* message from the L2GW. This implies that call-3 is considered active and the STB is considered in SERVICE-3 before the service connections are available. Figure 12 shows the other option, where the connections are established during the call set-up protocol. After the bandwidth needed for the connections is granted, the L2GW asks the video-server (in our example), or the STB, or both, to set up the required connections. When the connections are established, the L2GW is informed, and the call is considered active.

The main advantage of the first approach, where service connections are set up after the call set-up protocol terminates, is that the call set-up protocol lasts for a shorter period of time. This is important if key entities participating in this protocol, like the L2GW, need to complete the protocol before they are able to support other service requests. The second approach, where connections are set up during the call set-up protocol and are available when the SERVICE state is entered, has the following advantages. Firstly, if for any reason the connections cannot be established there is no need to invoke call-3 take-down protocol. Rather, the call-3 set-up protocol informs the STB that the call cannot be admitted. Secondly, this approach results in a more compact solution because it couples connections establishment with call signaling. However, the potential issue is the scalability, where the combined solution might be too complicated once the connection number is big.

The set-up timing and the layout of the *control connections* depend on the exact protocols used for set-up and take-down of the various session calls. For instance, in the call-2 set-up protocol considered in Figure 5(a), a bi-directional control connection is needed between the STB and the L1GW and between the L1GW and the L2GW. In the call-2 set-up protocol considered in Figure 5(b), a bi-directional control connection is needed only between the STB and the L2GW. In the first case, the control connection between the STB and the L1GW can be the same one used during the set up of call-1 (Figure 4). The latter connection has to be established when the STB moves from OFF to NO-SERVICE (see Figure 3). The control connection between the L1GW and the L2GW can be a semi-permanent one, established in advance for the exchange of all control messages between these two entities. If a control connection is needed between the STB and the L2GW, it should be set-up when the STB selects a L2GW in SERVICE-1 state.

Authentication and Authorization

In terms of the FSM model, authentication and authorization is a mechanism which ensures that transitions into some of the SERVICE states are performed only by authorized STBs and users. The distinction made here between an STB and a user is important: the access into some SERVICE states should be allowed to *all* the users operating *certain* (authorized) STBs, whereas the access into other SERVICE states should be allowed only to *certain* users operating *certain* STBs.

For instance, the transition from OFF into NO-SERVICE state can be controlled by an authentication mechanism which ensures that the STB is entitled for the VoD services. Later, when a transition into SERVICE-3 is required, an authentication mechanism can be invoked in order to ensure that the particular user operating the STB is allowed to watch the particular movie.

An authentication and authorization mechanism can be implemented in one of two ways. The first way is to view this mechanism as a service, associated with some SERVICE state. For instance, if SERVICE-3 state can be entered only from SERVICE-2 state, then authentication and authorization for call-3 can be performed during call-2 in SERVICE-2. If, however, SERVICE-3 could be entered from several states (e.g. SERVICE-2 and NO-SERVICE) then instead of integrating the authentication and authorization mechanism with each of the calls associated with these states, another SERVICE state, SERVICE-4 say, could be defined. In such a case a transition into SERVICE-3 would be allowed only from SERVICE-4, after the authentication and authorization call is successfully completed.

The other way to implement the authentication and authorization mechanism is to combine it with the call set-up protocol invoked before the SERVICE state is entered. An example for SERVICE-1 was introduced in Figure 5.

7 Conclusions

The paper was concerned with the provisioning of complex interactive services by means of network sessions. To this end, the paper considered a VoD network and showed that a VoD session can be modeled as a Finite-State-Machine (FSM) whose states represent the calls comprising the session. Each of the session calls provides the STB with some preliminary service, or with the selected video application stream. The transitions among states usually represent distributed protocols for set-up and take-down of calls.

The session FSM was used to illuminate and discuss the options and trade-offs involved in the design of a VoD session. The paper showed that the trade-offs depend upon the relationship among the various participating entities and provided services. The paper also presented possible specification for the call set-up and take-down protocols associated with the transitions among the session FSM states.

The FSM can be used in order to model even more complicated sessions. For instance, it is possible that the user will have to navigate through several L1GWs or L2GWs before getting the requested service. If domains of multiple L1GWs and/or multiple L2GWs are constructed hierarchically, more states and transitions should be defined.

Using the FSM model, a VoD session can be designed in the following way:

- Determine the states of the FSM. Each service should be represented by at least one state.
- Determine which call should be active and which calls should be alive in every state.
- Determine which states should be controlled by means of an authentication and authorization mechanism. This may result in some additional states.
- Determine which transitions should be allowed among the states.
- Determine when each application and control connection should be set up.
- Specify the call control protocols (set-up/take-down) associated with every transition.

8 References

[1] H. Appenzeller. Signaling system No. 7 ISDN user part. *IEEE Journal on Selected Areas in Communications*, 4(3):366–371, May 1986.

[2] The ATM Forum. ATM user-network interface specification, 1994.

[3] Bell Atlantic. Signaling Specification for Video Dial Tone, TR-72540, issue 1, release 2, December 1993.

[4] Y.H. Chang. Video Dial Tone (Level 1) Gateway, HP internal document, April 1994.

[5] Y.H. Chang, D. Coggins, D. Pitt, D. Skellern, M. Thapar, and C. Venkatraman. An Open-Systems Approach to Video on Demand, *IEEE Communications Magazine,* May 1994, pp. 68-80.

[6] Y.H. Chang and D. Pitt. Call Models for Video Dialtone, *International Switching Symposium* '95, Berlin, April 1995.

[7] D. Comer. *Internetworking With TCP/IP.* Prentice Hall, 1988.

[8] R. Cohen and A. Segall. Connection management and rerouting in ATM networks. In *INFOCOM*, 1994.

[9] DAVID's First Call for Proposals. version 1.0, Oct. 1994.

[10] Digital Storage Medium Command and Control (DSM CC), ISO/IEC 13818-1, Annex A, June 1994, pp. 73-82.

[11] Scientific Atlanta. Control System Messages for Interactive Video Services, Publication number 69L008A, Jan. 1994.

[12] S.Subramaniam, F. Aalei, S. Uppal and K. Rath. Session Signalling Protocol for Full Service Networks. ATM-Forum/94-1222, Nov. 1994.