Network Support for Turn-Taking in Multimedia Collaboration

Hans-Peter Dommel and J.J. Garcia-Luna-Aceves
Computer Engineering Department
University of California, Santa Cruz, CA 95064
E-mail: {peter, jj}@cse.ucsc.edu

ABSTRACT

The effectiveness of collaborative multimedia systems depends on the regulation of access to their shared resources, such as continuous media or instruments used concurrently by multiple parties. Existing applications use only simple protocols to mediate such resource contention. Their cooperative rules follow a strict agenda and are largely application-specific. The inherent problem of floor control lacks a systematic methodology. This paper presents a general model on floor control for correct, scalable, fine-grained and fair resource sharing that integrates user interaction with network conditions (Quality-of-Service), and adaptation to various media types. The notion of turn-taking known from psycholinguistics in studies on discourse structure is adapted for this framework. Viewed as a computational analogy to speech communication, online collaboration revolves around dynamically allocated access permissions called floors. The control semantics of floors derives from concurrency control methodology. An explicit specification and verification of a novel distributed Floor Control Protocol (FCP) are presented. Hosts assume sharing roles that allow for efficient dissemination of control information, agreeing on a floor holder which is granted mutually exclusive access to a resource. Performance analytic aspects of floor control protocols are also briefly discussed.
Keywords: floor control, multimedia collaboration, Quality-of-service adaptation, resource sharing, turn-taking.

1 INTRODUCTION

The advent of networked multimedia applications and the provision of tools for telecollaboration necessitates mechanisms to allow for mutually exclusive access and usage on shared resources, such as files, telepointers, continuous media channels, or instruments, which are jointly used at geographically dispersed stations in a computer network. This applies to both asynchronous deferred execution of operations on such resources,[19] as well as to synchronous, real-time interaction.[13] If two or more users concurrently operate on the same resource, race conditions and inconsistencies may occur. Under resource contention, shared work is prone for collisions on synchronously handled data, due to initiation or integration conflicts. The static side of these problems can be alleviated through fine-grained access control for collaboration.[27] Despite similarities between dynamic floor control on multimedia resources and database concurrency control, collaborative work can differ in many ways from prototypical advanced transaction processing models[2] or its extensions into the collaborative context.[1] New control protocols are hence needed to foster cooperation of varying scope among users and system processes on heterogeneous platforms.

Report Documentation Page		*Form Approved* *OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **1997**	2. REPORT TYPE	3. DATES COVERED **00-00-1997 to 00-00-1997**
4. TITLE AND SUBTITLE **Network Support for Turn-Taking in Multimedia Collaboration**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of California at Santa Cruz,Department of Computer Engineering,Santa Cruz,CA,95064**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES **12**	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**			

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

In this new generation of cooperative middleware, *floor control*[9] mediates in the interaction among humans or computational agents, in order to improve on sharing modalities in the alternation between private and individual activity vs. public and group-oriented workmodes. Floor control can build on admission-control decisions[12] in granting access to network resources. The term "floor", in its original connotation, refers to the right to speak in the conduction of formal meetings.[23] We define a *(computational) floor* as an ephemeral permission, placed on shared resources by a distributed floor control protocol to mediate user interactions on those resources. While other collaborative services, such as IP multicasting, real-time transport protocols or connection control protocols for multimedia conferencing[13] receive a fair amount of attention, conflict and competition in cooperation and their resolution have not been treated previously in a wider context.

While current distributed collaborative systems mostly allow only for unimodal sharing of few resources in small sessions, the usage of multiple resource types in collaboration introduces many new requirements for floor control. Critical application states must be replicated and changes be sent reliably and conformally across a variety of network links with different Quality-of-service (QoS), and certain media are codependent and must be delivered reliably and synchronously. Floor control has to perform presentation control for processing functions on shared objects, quality control for adjusting the characteristics of resource utilization (e.g., with QoS parameters such as resolution, frame-rate, etc.), activity control for monitoring conditions by which floors are to be triggered or revoked (e.g., video frame boundaries, speaker interruption etc.), and synchronization control for inter- and intra-flow connection of resource objects in timed presentation and usage, e.g., in the coupling of video and audio.

Floor control has been previously associated primarily with "tightly coupled" conferences where group membership is explicitly known. However, there are cases where floor control is also necessary for "light-weight sessions"[8] which lack explicit membership registration, such as in the remote multiparty control of shared objects or devices through the World Wide Web. While several individual algorithmic solutions for floor control have been proposed together with experimental implementations of networked multimedia systems,[7] an underlying development methodology and general framework are still lacking. In particular, no explicit description of a floor control algorithm can be found in the literature, the reported solutions are often cast into a specific application and architecture, and in most cases floor management solutions are too simplistic in order to scale, or guarantee reliability and responsiveness.[6]

The focus of this paper are the various aspects of multiparty interaction and its distributed control. We retrace the problem of resource sharing to interaction models known from psycholinguistics, specifically to *turn-taking* applied in the analysis of conversations.[22] Our intention is to provide a schematic foundation for floor control protocols that observes both technical aspects such as network conditions and mixed media bindings, as well as user interaction behavior, in order to better understand the nature of turn-taking in resource sharing within networked multimedia environments. Section 2 presents a general formal model of turn-taking, discussing time and causality in the control of sharing activities. Section 3 presents principal floor control concepts such as adaptation to various media and Quality-of-Service conditions in the network. Section 4 specifies a general floor control protocol and verifies its correctness. Some considerations on the performance evaluation of floor control protocols are also outlined. The paper concludes in Section 5.

2   TURNTAKING REFERENCE MODEL

Cooperation as a sequence of activities on shared information bears an inherent structure resembling a discourse model as it is known from linguistic pragmatics.[17] A central concept in the analysis of conversational flow, with focus on interactional patterns based on ethnic production and interpretation of social interaction, is *turn-taking*,[24] which is defined as the passing of speaker control among multiple parties. This notion of turn-taking has its parallels in the domain of multiparty interaction via networked computers, as it is studied in computer-mediated communication (CMC)[21] or computer-supported cooperative work (CSCW). Grice's interaction maxims,[11] which comprise quantity (as little as possible, as much as necessary), quality (truthfulness),

relation (relevance), and manner (clarity), are more obeyed in computer-mediated interaction than in face-to-face meetings.[14] While floor control is primarily targeted at cooperation among humans, it can also be applied to resource negotiation in agent encounters,[5] where both initiative and reactiveness must be reflected in the control strategy. The integrity of a session is based on the mutual consent of all participants on membership and floor holding terms. In this respect, cooperation entails agreement on the control aspect (form) as well as the shared work itself (content).

The notion of a conversational floor can hence be generalized to multimodal interaction via a computational floor responsible for regulating concurrent activities on shared multimedia resources. The introduction of such control elements to online collaboration is important, since remote cooperation cannot rely on exterior awareness cues to signal the passing of control, as it is possible in face-to-face encounters with eye-gaze or other social signaling. Conversational turn-taking without further mediation works well between two or three parties in the absence of visual monitoring, e.g., via telephone, where only 5% of alternating speech is subject to overlap,[17] equal to the overlap ratio in face-to-face meetings. In telecollaboration, explicit control protocols must compensate for the lack of social embedding, prevent activity overlap and hence conflicts, and allow for various mechanisms in the transfer of controllership and provision of service policies. The significance of floor control becomes apparent for large sessions (more than 12 participants) and many concurrently shared resources. While turn-taking theory also includes management of "adjacency pairs" of action-reaction activities, repair, and undoing, our focus is the basic provision of correct and efficient turn-taking control.

In pragmatic turn-taking models, at each turn a party assumes a social role such as "speaker" or "listener", switching control under the premise of minimizing pauses and maximizing the conveyed information. For instance, among two people L ("Local") and R ("Remote") a distribution pattern such as L-R-L-R-L-R... can be observed for the speaker floor, with varying length of holding times. The concept of *transition relevance points* (TRP) was introduced by Sacks et.al.[24] to aid in the dissection of interactive turns. At each TRP, identified by syntactical constructs, control may eventually be switched. The challenge in designing floor control mechanisms is to achieve similar smooth transitions[30] for telecollaborative work, replacing the notion of TRPs with other control rules adhering to the usage semantics of various media.

Designing control mechanisms for distributed work gives choices in how controllership is performed. Control addresses not only the joint usage of resources, but also the management of data replication and coherency preservation. Multilateral control can be implemented with varying degrees of distribution, from fully centralized to fully distributed realization. It can be successive (only one distinct controller at a time, who is fixed or roving), partitioned (several controllers, each performing a subset of control operations), democratic (all participants contribute to the control process, e.g., via voting), or anarchic (participants retain complete freedom of acting through peer-to-peer sharing of control). Other design decisions include whether control behavior is defined a priori or negotiated *ad hoc*, and how complete each node's observations on the global system state are. A shared workspace constantly reshapes itself in the progress of group efforts. Well-managed control in the sense of mediation, which adapts to the workspace dynamics, may significantly improve the quality of collaboration. For our model we assume successive controllership.

We outline a floor control model to formally characterize the passing of control over shared resources at all nodes. Users and their cooperative processes are identified with nodes in a graph. We define a collaborative environment $\mathcal{C}$ as a pair (N, E) with a node set N and $E \subset N \times N$ as a set of edges representing connections. A connection is a unidirectional or bidirectional transmission link from a sender node to a set of receiver nodes. Each node connects selectively with other nodes in a session, placing its public resources into $\mathcal{C}$. The floor control protocol runs in dedicated hosts responsible for managing specific resources. The floor is the scarce resource, around which turns and resource usage revolve. Each control protocol performs at the same time server and client operations, reflecting the alternation of user roles as a contributor or observer.

Figure 1 depicts a two-party interaction, with model (a) describing the turn-taking flow[20] and model (b) as its corresponding generalization to a general collaborative control, with the notation Lx_iR, $L, R \in N$ hosts, x_i
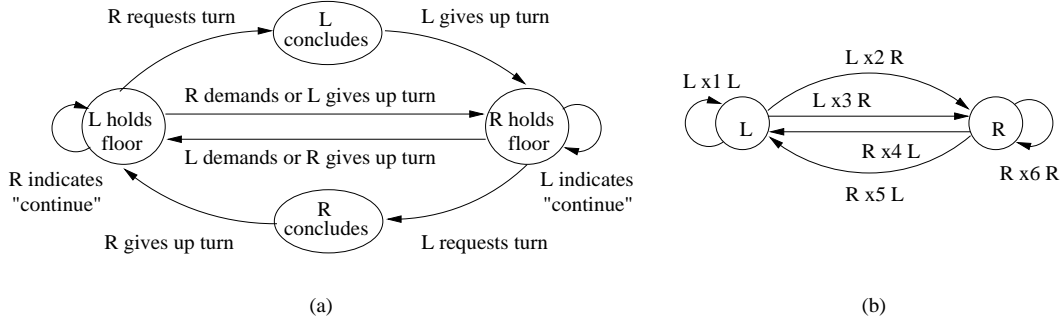
Figure 1: (a) Turn-taking flow; (b) Control and activity flow abstraction.

as a symmetric relation between the local and remote host, i a natural number, defining an *activity*. A two-party model is a sufficient abstraction, since any multiparty collaboration among N nodes can be abstracted into maximally $\frac{N(N-1)}{2}$ pairwise interactions. Model 1(b) has two interpretations:

1. The *control flow* describes floor passing: x_1 symbolizing "L (continually) holds the floor", x_2 is "L concludes and relinquishes floor to requesting R", x_3 is "floor revoked from L and handed to R". Interaction is symmetric and $x_i, i = 4, 5, 6$ represent the reverse cases.

2. The *activity flow* on resources R_j, with $x_i \in R_j$, j denoting a specific resource-type: for example, $R_{voice} = \{talk, listen, mute, replay, \ldots\}$ represents a set of operations on the audio channel, and $x_1 = talk$ depicts L having the floor for talking to R. Certain activities may be forbidden or void in a given directionality i, depending on user authorizations and resource types.

Turn-taking may be locally managed on a turn-by-turn basis, or follow a more strict group-wide agenda, depending on the session format. Cooperation unites antagonist forces, namely the urge to maximize individual resource utilization ("self-interest") vs. fostering cooperation among participants ("group-interest"), which defines various fairness conditions for floor assignment. Fairness is essentially a product of the underlying distributive mechanism for floors, corresponding to the transport mechanism and a specific floor policy.[4] A policy[26] is implemented by a specific ordering in the service of requests, together with revocation conditions and timeouts. It may support queueing and fairness conditions through request reordering, deferred execution, and preemption of activities. Certain mechanism limit the choice of policies, such as token passing which assumes by default a Round-Robin policy. Policies can also include priorities or "floor credits" computed from floor usage statistics (frequency of access, cumulative duration etc.) in the collaborative history. Such credits increase with waiting time for the floor, or decrease at a rate reflecting an activity's cost. Flexible policies allow to adapt control to the *ad hoc* dynamics of turn-taking by observing current network conditions and node capabilities.

We briefly characterize turn-taking with respect to causality of activities (operations on shared resources) and the timeline of floor usage. Combining floor assignment with causality conditions allows to characterize collaboration formally with a partial precedence relation to indicate separateness or conflict freedom on a shared resource. Collaborative work can hence be viewed as a sequence of causally related activities, each intrinsic to a specific turn. This defines a mapping α from the set of activities A, as a finite series of cooperative tasks performed on specific resources R, to the set of turns T:

$$\alpha : A \rightarrow_R T$$

Actual representations of activities in users' shared workspaces are joint operations on shared information, or streams transmitting media packets between sites. Each activity can have an *outcome o* in the shared workspace,

as a side-effect of an activity. Two activities $a_1, a_2 \in A$ are partially ordered by

$$a_1 \leq_c a_2$$

which signifies, by means of the causal precedence relation $\leq_c$ ("happens before"[16]) that a_1 is causal for a_2, as an episode in the logical continuation of the shared work process. Floor assignment must observe such causal precedence of activities, characterized by one or more of the following conditions:

1. Activity a_1 happens before activity a_2;

2. Activity a_2 uses an outcome o_1 of activity a_1;

3. Activity a_2 starts after activity a_1 has finished (temporal causality);

4. There exists an activity *sequence* $a_{i,1}, a_{i,2}, \ldots a_{i,l}$ such that $a_{i,1} = a_1, a_{i,l} = a_2$, and for all $j \in [1, l-1]$, the activities $a_{i,j}$ and $a_{i,j+1}$ are either causally related by 1. - 3.

Condition 3. implies 1., but not vice versa. An activity is called *sequential*, if the causal ordering relation $\leq_c$ is linear. If a_1 and a_2 do not causally precede each other, they are called *independent* and may be carried out concurrently without floor-controlled mediation. Viewing activities as building blocks for executing tasks, causal and temporal precedence and their impact on floor controlled sharing also apply to subsets of a sequence ("partial activity") and sequences of sequences ("composite activity"). Activities can be either goal-oriented, e.g., finishing by a specific deadline, or process oriented, e.g., by following a given agenda, but without goal-orientation.

A turn, as a sequence $T = (a_1, a_2, \ldots, a_i)(r_j)$ of activities a on a resource RJ, must guarantee atomicity of the performed operation. Its 'floor lifecycle" includes signaling and usage from onset to handoff. A turn is called *correct* if the floor management governing the turn is correct. For a series of turns, $(T_1, T_2, \ldots, T_k)$, each with varying duration and governed by a distributively allocated floor, the partial order $(\Sigma, <_H)$ defines a *collaborative history*, with Σ as the set of all operations executed during turns T_i, and $<_H$ totally ordering all conflicting operations. A floor control protocol must ensure that all turns in the collaborative histories are correct.
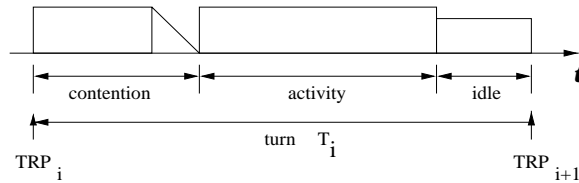

Figure 2: Conceptual model for turn structure.

The timeline of a turn for a given resource is shown in Figure 2. It consists of a floor contention period, an activity period, and a floor idle period. Handover, marking TRPs, can be incited through activity completion, timeout, or a floor controller decision to revoke the floor. Idle periods in a turn can be used by another node for brief feedback without revoking the floor from the current holder.

3   RESOURCE-ADAPTIVE FLOOR CONTROL

A protocol has to observe user behavior, application and host states, and network conditions in the floor allocation process. In particular, acquisition rules should correlate the realizable vs. demanded QoS, which is measured for different applications with different goal functions, such as end-to-end delay, quality, error rate,

and lossiness. The metric is a cost function that allows to cross-relate and combine different QoS parameters in order to optimize the delivery of joint multimedia streams. For example, for specific nodes, which cooperate with video and audio, different bandwidth and link characteristics create fluctuating qualities for video (frame-rate, jitter) and audio (sample-rate, packet loss rate). At each turn, floor allocation can take current admission control measurements into account, allocating the floor to the node that yields the best possible channel utilization. The goal is to optimize turn-taking by maximizing the QoS in each activity, turn, and sequence of turns. For that purpose, a collaborative resource semantics must be defined, with H denoting the current floor holder, N as the next holder in line, and TRP as a transition relevance point for floor handover:

1. Rules applying to the first TRP of any turn:

    (a) N must obtain a floor on a shared resource before executing an operation on that resource.

    (b) If H selects N for next turn, H must relinquish the floor and N takes the floor in the follow-up TRP.

    (c) If H does not select N, then any other party may self-select, with the floor being granted to the first person according to the policy in use (by default First-Come-First-Served).

    (d) If H has not selected N, and no other party self-selects, then H may (but need not) continue and claim the floor for the next turn-constructional unit.

2. Rules applying to subsequent TRPs:

    (a) Rules 1 (a) to (c) apply recursively to the next TRP to determine the next floor holder in subsequent turns.

    (b) On a turn and for any two operations p and q, if the floor for p is compatible with the floor for q, the floor for q may be acquired after the completion of p, such that q observes the effects of p.

    (c) Multiple instances of floors for the same resource R_j may only be assigned to node sets, which are disjoint, i.e., for two floor holders H_1 and H_2 and node sets N_m and N_n, $(H_1 \cup N_m) \cap (H_2 \cup N_n) = \emptyset$, $m \neq n \leq |N|$.

    (d) Codependent floors (for synchronized media such as audio and video) must be acquired and released together.

A floor control protocol, executing a mutual exclusion algorithm,[28] operates on the following premises:

- The session node set of size N may expand or shrink dynamically due to session splits, merges, joins and leaves of member hosts.

- M hosts, $M \leq N$, may acquire the floor for one resource and enter the critical section of resource access, if the resource semantics permits it (e.g., with telepointers).

- K hosts, $K \leq N$, may fail in a session without affecting the correctness of the floor control algorithm, which is then called K-resilient.

The number of floors to be controlled depends on the object structure of the resource, its sharing semantics and granularity level. For a session with N nodes, R_u ubiquitous resources, R_l local resources, and R_m mediating resources, each with maximal sharing granularity g, an upper bound on the floors F to be tracked distributively by a control protocol is

$$F \leq N(g_u R_u + g_l R_l) + \frac{N(N-1)}{2} g_m R_m \tag{1}$$

If media are captured or presented together, e.g., for audio and video, such resource bindings for the same level of granularity are captured by *codependency floors*. Resource containment is reflected in *hierarchical floors*, which allows for increased granularity. Before a child granule is granted a floor, an intentional floor (i-floor) needs to be placed on its parent resource, similar to hierarchical locking schemes. However, the resource semantics for floor allocation differs from locking, since it is dynamic and multimodal. Two operations conflict if they operate on the same granule and if both of them permanently affect the current state of the granule. For example, the `listen` operation, resembling a "read" operation on files, does not change the status of an audio channel, whereas `talk`, equaling "write", changes the status. Similar to lock compatibility matrices in database theory,[10] floor transitions need to be performed according to a floor compatibility matrix. Valid operations for each resource are derived from its collaborative semantics.

Requested Mode	Established Mode						
	none	**L**	**T**	**H**	**I**	**Ml**	**Mr**
L	○						○
T			●	●	○	●	
H	○				○		
I	○		○		●	●	
Ml			●		●		
Mr	○	●			○		

Table 1: Compatibility between audio floor modes.

An example floor matrix for audio communication is shown in Table 1. ○ indicates a weak conflict, which does not necessarily impede joint work, and ● indicates a strong conflict, which creates inconsistencies or mutual blocking in the work progress. In the table, **L** denotes listen, **T** denotes "talk", **H** denotes "hold", **I** denotes "interject", **Ml** denotes "mute-local" on a microphone, and **Mr** denotes a "mute-remote" at the source or on audio speakers. Similar floor compatibilities can be defined for other multimedia resource types, such as for video or concurrent manipulation of operation modes in a remotely controlled instrument.

We briefly discuss, how QoS tuning under multiple constraints can be merged with floor control. The QoS for each activity can be characterized as a vector, whose components depict service quality conditions. QoS vectors $QoS_{resource} = (c_1, c_2, \ldots, c_l)$ are of equal dimension for all multimedia resources, however, depending on the applicability of a characteristic c_i, i a natural number, the vector component is instantiated, or null-valued otherwise. A sample vector for audio is $QoS_{audio} = (samplerate, samplesize, lossrate)$. To evaluate a resource specific QoS, the actual values for the characteristics c_i are biased, within an allowable [best-effort, optimal] service interval, with weights w_i, i a natural number, and a normalization factor n_i, which correlates the expense in transmitting packets for the resource across the network, yielding the *characteristic QoS* (*cQoS*):

$$cQoS_{resource} = \sum_{i=1}^{l} w_i c_i n_i$$

This allows for a normed representation of different QoS for all types of resources and fast evaluation of QoS conditions. Determination of the weights is resource-dependent. To allow the user tuning of QoS values, the *feasible* QoS, given with $cQoS_f$, is evaluated against the *demanded* $cQoS_d$. Ideally, $cQoS_d$ approximates $cQoS_f$ closely. For multiparty interaction, is it necessary to percolate QoS parameters to the application layer, in order to let users directly influence resource usage. To incorporate floor allocation in the *media scaling*[29] process, we define an average individual floor allocation function F_{alloc} for a node X as

$$F_{alloc}^{X} = (\sum_{i=1}^{s} nT_i dT_i - \bar{w}T_i)/N$$

with s as the cumulative number of turns in the session, nT_k denoting the number of previous turns, dT_{il} denoting the duration at each turn l, and $\bar{w}T_i$ as the average number of turns that the node has to wait to acquire the floor. At each turn and during the contention period for a floor, the *floor evaluation* function η takes the $cQoS$ and F_{alloc} with a priority value p into account to determine the node with the highest value as the next floor holder:

$$\eta = (F_{alloc} + cQoS_{resource})p$$

This heuristic scheme is not intended to influence network-layer admission procedures, but rather to incorporate knowledge about network conditions into application-level interaction and resource usage. The challenge lies in finding the correct parameterization of QoS vectors.

4 FCP: FLOOR CONTROL PROTOCOL

4.1 Protocol Definition

We discuss now a Floor Control Protocol (FCP) that adheres to the previous considerations. It also allows for queueing of requests and variations in the service policy. To our knowledge, this is the first explicit specification of such a protocol in the literature. The protocol is fully distributed and is assumed to run independently at each host participating in a collaborative session. Opposite to other existing protocol schemes,[7] the protocol is resource-adaptive, and embodies a subset of *control roles* $\mathcal{CR}$, which can be interpreted as the dynamic counterpart to static role-based access control.[25] The roles depend on the nature of the shared resource, which is either ubiquitous and hence available at multiple sites (e.g., files), localized (e.g., an instrument, which is attached to one site, but whose functions can be controlled remotely), or mediating between several hosts (e.g., an audio-channel). Contrary to other control paradigms[7] such as token-passing or activity sensing, FCP does not presume a predefined hop sequence or rely on a short propagation delay between nodes.

Formally defined as a triplet $\mathcal{CR} = \{O, C, H\}$, the floor owner O is unique for each shared resource and introduces, owns, and withdraws it in the collaborative workspace. The floor coordinator C, one per resource, is the principal process regulating who may attain which floor at what time. The floor holder H is the current temporary user of the resource governed by that floor. At the beginning of a session, floor ownership and holdership are decided depending on the joining order of nodes to the session. For a resource R_j introduced first by node N_X, this node is by default O_j, C_j, and H_j, until distributed control shifts these positions in the course of collaboration. There may be several floor holders per resource, if the usage semantics of that resource allows for multiple users under mutual agreement, e.g. in the case of multiple graphical cursors in a shared editing system. Information on roles is transferred with each control packet, i.e., each active site is informed at all times about the current overall control state of the collaborative system. Hosts, users, and resources are assumed to be uniquely identified within a session. FCP interfaces with a session directory that keeps track of the current organization of the session, services requests, and tracks floors for local resources and remote operations. Figure 3 shows a specification of FCP in the form of a state machine.

STATES

IDLE	IDLE (no activity)
LHF	Locally-Held-Floor
LRF	Locally-Requested-Floor
RHF	Remotely-Held-Floor
RRF	Remotely-Requested-Floor

ACTIONS / EVENTS

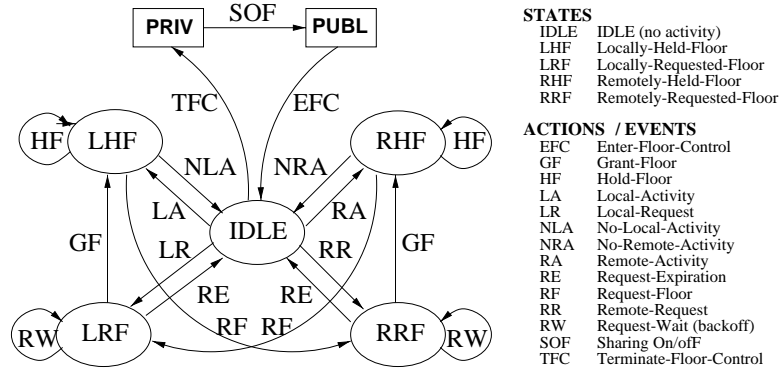|	
EFC	Enter-Floor-Control
GF	Grant-Floor
HF	Hold-Floor
LA	Local-Activity
LR	Local-Request
NLA	No-Local-Activity
NRA	No-Remote-Activity
RA	Remote-Activity
RE	Request-Expiration
RF	Request-Floor
RR	Remote-Request
RW	Request-Wait (backoff)
SOF	Sharing On/ofF
TFC	Terminate-Floor-Control

Figure 3: FCP protocol state diagram.

The mutual exclusion algorithm in FCP does not presume a prescribed topology. The protocol states, transitory actions and events are traced in as many instances as there are floors to track. Control state information is disseminated between nodes only when they interact. Each node keeps a local and partially replicated state table, which records the current state of the floor distribution. Table management is weakly consistent, receiving updates for a specific floor only, if its node is active. Control packets contain per-turn information on the sender and receiver nodes, session, user, and floor state, which comprises the resource, activity, and η. The distinction between local and remote control, reflecting the symmetric turn-taking scheme described earlier, allows separation of local and remote affairs at each node and tracking of multiple activities on the same resource type among

several partitions of the node set. The protocol consists of five main states (LHF, RHF, IDLE, LRF, RRF) to represent submission of floor requests, attempts to claim a floor, actual floor usage, idleness and release, both for the floor holder H floor coordinator and C. The control operations executed by H are **attempt, acquire, use, release, withdraw**, and C executes **elect, defer, grant**, and **revoke**. The state diagram is simplified in that it does not contain transitions for feedback floors or exceptions such as node failures, which necessitates floor recreation and role elections among the remaining nodes. We assume that sharing for each resource can be switched on or off by the local owner of a resource (SOF). Once a resource is declared public, FCP by default assumes no activity (EFC into IDLE). In an IDLE state, C detects no messages from either local or remote users. Either the local or remote node issues a floor request (LR, RF), inciting a transition into a wait state (LRF, RRF), from which floor claiming is attempted. Nodes can wait for a specified time (RW), after which the request expires (RE) or the floor is granted (GF). Once the local or remote node acquires the floor as H and uses the resource (LHF, RHF), it can continue to do so (HF) until a floor request from a remote node is registered (RF), shifting the protocol to a wait state for the other node. All request, wait, and hold actions are timed, or can be revoked preemptively by C and observe the turn-taking rules described. The time interval that C takes to decide upon the succeeding floor holder must be large enough to allow every other potential H to send requests to this site. Enforcing acknowledgement on floor migration implements atomicity of the floor state change. As long as an idle floor is claimed or in migration to another node, it is attributed to the previous H. If a migration fails, the previous H is the default recovery location.

While FCP does not presume a specific logical topology among nodes, an underlying multicast protocol[31] may impose a specific delivery strategy on the sending and receiving nodes in a session. Resource usage is, depending on the floors granted to other sites for that resource, multicast to all members of the multicast group collaborating on the resource. In case of simultaneous claims for the same floor, the race condition is resolved by the request packet arrival order and floor sequence index, or permuted based on an agreed-upon floor policy. State transitions and timers are attuned to specific media characteristics. Multiple incoming floor requests may or may not be queued, depending on the resource and service policy. Without queueing, unsuccessful users must retry until they get the desired floor. In order to render floor management unobtrusive and integrated with operations on shared resources, control must be anchored within the resource handling semantics, e.g., in form of voice-activated floor allocation for audio conferencing. Temporary withdrawal or leaving of a session terminates floor control (TFC) for the resources from the leaving host, with the option to restore the previous state in the case of rejoining from a log file that FCP maintains throughout a session. To minimize waiting time on floor requests, a RF discard strategy can be implemented with limited queueing, if too many RFs are received at a floor server. Fairness of FCP is resource-specific and depends on the established service policy.

4.2 Correctness of FCP

We show that FCP guarantees safety and liveness. More specifically, if a resource allows K concurrent floor holders, FCP permits at most K nodes to access the resource at any time, and every floor request is serviced within finite time. We assume failure-free communication in the network and a bounded end-to-end delay between nodes. Floor transitions are acknowledged from the new floor holder H to the floor coordinator C. The following arguments assume a unique and totally ordered indexing scheme among the nodes and activities, atomicity in the transitions between protocol states, and that no code segment associated with any event in the protocol can delay execution.

Theorem: FCP is safe.
Proof: We have to show that mutual exclusion is achieved. By contradiction, for K nodes being granted a floor, $M > K$ nodes actually have the floor. Floors F_{ji} for a resource R_j are indexed with labels $i = 1, \ldots, M$, defining an order $(F_{j1}, N_1) \leq (F_{jK}, N_K) \leq (F_{jK+1}, N_{K+1}) \leq (F_{jM}, N_M)$. In order to attain a floor, node N_{K+1} must have received a GF message from the current C_j. There are only three possible cases in which C_j may have received a request message RF. (1) A node $N_X, X \in 1, \ldots K$, was already attempting to gain the floor from C_j with floor sequence number F_{jX}; in this case, C_j would then have deferred the request from N_{K+1}. (2) Node N_K

was already operating on the resource in a previous turn; in this case, C_j would not grant another floor, until one of the nodes $N_1, \ldots, N_K$ releases at least one floor instance, since $F_{jK+1} > F_{jK}$. (3) At least one of the nodes $N_X, X \in 1, \ldots, K$, is IDLE such that its floor is revoked after a timeout; in this case, request F_{jK+1} is mapped onto F_{jX} and hence node N_{K+1} can never attain a floor, which contradicts the assumption. $\square$

Theorem: FCP is live.

Proof: We have to show that (a) no deadlocks occur, and (b) no livelocks (starvations) occur. Ad (a): A deadlock happens when fewer than K nodes hold a floor and a RF message from node N_X is unserviced. Suppose that a deadlock at turn T occurred and the collaboration is deadlocked. Using the previously employed indexing scheme on floors, a RF can be deferred indefinitely only if either C_j or H_j defer to reply to N_X. If C_j does not respond within timeout, a new controller C'_j for R_j is elected. If the current floor holder H_j, with $1 \leq \ldots \leq H_j \leq \ldots \leq X \leq \ldots \leq K$, does not reply, the floor is revoked from H_j after a timeout and is assigned to node N_X. Therefore, at least one of the $N - K$ nodes that does not currently hold a floor, eventually receives the floor, which is a contradiction. Ad (b): A node is said to starve, when its RF message is indefinitely deferred while other nodes are being served. Consider node N_X trying to attain floor F_j by sending a RF to C_j. This request may be immediately serviced, or deferred if another node N_Y holds this floor, or is attempting to do so with a lower request index $h < j$. At most $j - h$ other requests may be served before R_j is served; therefore, $N_X = H'_j$ eventually acquires the floor F_{jX}. $\square$

4.3 Performance Considerations

The following questions arise with regard to floor control protocol analysis: how does a collaborative system scale up in terms of "collaborative throughput" and response time, since intelligent scheduling of resource access can improve interaction flow; what is the appropriate level of sharing granularity for various media; what are the effects of waiting periods, queueing of floor requests, and aborts of floor controlled operations; which control mechanism and allocation policy for the dissemination of floor requests and grants performs best under certain network states; and how do network conditions and interaction patterns for specific media influence each other? Important analysis parameters are the session scale, the number of resources and granules, the floor request interarrival rate, the mean turn length, the mean number of wait cycles for a node to attain a floor, its frequency of attaining a floor, and the number of concurrent floors allowed for a resource. We will briefly address some of the above questions.

Throughput in resource access can be defined as the average successful floor utilization, divided by the average turn length which consists of a busy period, where the resource is tried to be accessed, and an idle period without activity (cf. Figure 2). The control message cost M_T for a complete turn T consists of the cost m for disseminating a floor request, allocation of a floor and adjunct resource, floor release and state table update:

$$M_T = m_{request} + m_{grant} + m_{release} + m_{update} \tag{2}$$

The initial C_j broadcasts floor information on resource R_j to $N - 1$ other nodes. For node N_X to acquire a floor, it must send one RF message to C_j, which defers this request, if the floor is busy, or replies immediately by sending one GF message to N_X, and $N - 2$ messages (or less) identifying the new H'_j to the remaining nodes, depending on which ones are engaged and active in collaboration on R_j. In the case of floor revocation, the messaging cost is one preemptive RF sent to H_j, which acknowledges release of the floor, one message to the new H'_j, and an update broadcast to the remainder node set, yielding an average of $M_T = 3N - 1 = O(N)$ messages. For certain sessions, floor transactions do not necessitate a controller C (such as a chairperson), and can be conducted by keeping track of the current floor holder H only. The actual algorithmic cost may vary, depending on the underlying logical node topology in the mutual exclusion algorithm used to find consensus on floor holdership. Control message multicasting can further reduce the cost of state information dissemination. With a resource access time αt, a floor request arrival at every α seconds (arrival rate $\lambda = \frac{1}{\alpha}$), and a processing overhead p per turn T, the number F_p of pending floor requests ("customers in the system") in a closed queueing

scenario is according to Little's Law[3]

$$F_p = \lambda T = t + \frac{p}{\alpha} \tag{3}$$

In the case that floor requests are queued, we would like to obtain a lower bound on the probability β that a floor request is denied. For N users requesting one of K floors, and with an average waiting time $\bar{w}$, we have $K = \lambda \bar{w}$ and hence $T = \frac{\bar{w}}{K} N$. Let β be the proportion of customers whose floor requests are denied. Since not all resources may be accessible, let $\bar{K}$ denote the number or busy floors. Then $\bar{K} = (1 - \beta)\, \lambda \bar{w}$ and since $\bar{K} \leq K$,

$$\beta \geq 1 - \frac{K}{\lambda \bar{w}} \tag{4}$$

5   CONCLUSION

While telecollaboration technology has displayed much progress since its inception,[15] as many existing collaborative applications[18] demonstrate, progress in networking and multimedia technology, together with more complex shared applications and larger collaborative sessions incites a need for more sophisticated cooperative protocols for highly interactive multiparty resource sharing. The turn-taking paradigm, as a metaphor for social protocols used in face-to-face interaction, is a simple structuring method to describe the flow of control in collaborative work. The floor control protocol FCP described in this paper is based on a turn-taking model, which allows to symmetrically capture and control sharing activities at geographically dispersed hosts. It differs from previous protocols in various aspects: its role-based floor management allows to accommodate for both a chairperson or completely automatic control of resource access and reduces the messaging cost to decide on a new floor controller; the protocol specification adheres to the hypothesis that different media can be serviced uniformly under one resource-adaptive control scheme; control can be centralized or fully distributed, depending on the resource; and the protocol does not presume a specific topology and service order for hosts, reflecting the spontaneous handover of control in collaboration. The protocol is currently used in multiparty access of a remotely controllable camera as a centralized example, and for distributed collaborative applications using audio and video. We continue our research with focus on comparative performance studies of floor control protocols, analysis of interaction patterns and QoS adaptation for various media, and development of further test applications. The concepts discussed in this paper are also applicable to nomadic collaboration.

ACKNOWLEDGEMENTS

6   REFERENCES

[1] D. Agrawal, J.L. Bruno, A. El Abbadi, and V. Krishnaswamy. Managing concurrent activities in collaborative environments. Technical Report TRCS94-05, UCSB, Santa Barbara, CA, 1994.

[2] N.S. Barghouti and G.E. Kaiser. Concurrency control in advanced database applications. *ACM Computing Surveys*, 23(3):269–317, September 1991.

[3] D. Bertsekas and R. Gallager. *Data networks*. Prentice Hall, Englewood Cliffs, N.J., 2nd ed. edition, 1992.

[4] T. Crowley, P. Milazzo, E. Baker, H. Forsdick, and R. Tomlinson. MMConf: An infrastructure for building shared multimedia applications. In *Proc. CSCW'90*, pages 637–650, Los Angeles, CA, October 1990. ACM Press, New York, NY.

[5] K. Decker and V.R. Lesser. Coordination assistance for mixed human and computational agent systems. Technical Report TR 95-31, Univ. of Mass., Amherst, MA, May 1995.

[6] H.-P. Dommel and J.J. Garcia-Luna-Aceves. Design issues for floor control protocols. In *Proc. Multimedia Computing and Networking '95*, pages 305–16, San Jose, CA, February 1995. IS&T SPIE.

[7] H.-P. Dommel and J.J. Garcia-Luna-Aceves. Floor control for multimedia conferencing and collaboration. *Multimedia Systems (ACM/Springer)*, 5(1), January 1997.

[8] S. Floyd, V. Jacobson, S. McCanne, C.-G. Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level-framing. In *Proc. Sigcomm'95, URL ftp://ftp.ee.lbl.gov/papers/srm_sigcomm.ps.Z*, pages 342–356, Cambridge, MA, August/September 1995 1995. ACM.

[9] J.J. Garcia-Luna, E. Craighill, and R. Lang. Floor management and control for multimedia conferencing. In *Proc. IEEE Multimedia '89, 2nd COMSOC Int. Multimedia Communications Workshop*, Ottawa, Canada, April 1989.

[10] J.N. Gray, R.A. Lorie, G.R. Putzolu, and I.L. Traiger. Granularity of locks and degrees of consistency in a shared data base. In *Modelling in Data Base Management Systems*. North-Holland, 1976.

[11] H.P. Grice. Logic and conversation. In *P. Cole and J. Morgan (Eds.) Syntax and Semantics 3: Speech Acts*. Academic Press, 1975.

[12] A. Gupta, W. Howe, M. Moran, and Q. Nguyen. Resource sharing for multi-party real-time communications. In *Proc. Infocom*, Boston, MA, April 1995. IEEE.

[13] M. Handley and J. Crowcroft. The Internet multimedia conferencing architecture. *ConneXions - The Interoperability Report*, 10(6):2–13, June 1996.

[14] E.A. Isaacs, T. Morris, T.K. Rodriguez, and J.C. Tang. A comparison of face-to-face and distributed presentations. In *Proc. CHI'95*, Denver, CO, May 1995. ACM SIGCHI.

[15] Bell Research Labs. The PicturePhone system. *Bell Systems Technical Journal*, Feb. 1971.

[16] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Comm. ACM*, 21(7):558–565, 1978.

[17] S.C. Levinson. *Pragmatics*. Textbooks in Linguistics. Cambridge University Press, Cambridge, UK, 1983.

[18] P.S. Malm. The unOfficial Yellow Pages of CSCW - groupware, prototypes and projects. Technical report, University of Tromsø, Norway, January 1994. URL *http://www11.informatik.tu-muenchen.de/cscw/yp/*.

[19] N. R. Manohar and A. Prakash. A flexible architecture for integrating heterogeneous replayable workspaces. In *Proc. Third Int'l Conf on Multimedia Computing and Systems*, pages 274–278, Hiroshima, Japan, June 1996. IEEE.

[20] A. McKinlay, R. Procter, O. Masting, R. Woodburn, and J. Arnott. Studies of turn-taking in computer-mediated communications. *Interacting with Computers*, 6(2):151–171, June 1994.

[21] D.G. Novick and J. Walpole. Enhancing the efficiency of multiparty interaction through computer mediation. *Interacting with computers*, 2(2):227–246, Aug. 1990.

[22] D.C. O'Connell, S. Kowal, and E. Kaltenbacher. Turn-taking: A critical analysis of the research tradition. *Journal of Psycholinguistic Research*, 19(6):345–373, Nov. 1990.

[23] H. M. Robert. *Robert's rules of order*. Bantam Books, Toronto; New York, 1986, c1982.

[24] H. Sacks, E.A. Schlegloff, and G. Jefferson. A simplest systematics for the organization of turn-taking for conversations. *Language*, 50(4):696–735, 1974.

[25] R.S. Sandhu and E.J. Coyne. Role-based access control models. *Computer*, 29(2):38–47, Feb. 1996.

[26] H. Schulzrinne, J. Kurose, and D. Towsley. An evaluation of scheduling mechanisms for providing best-effort, real-time communication in wide-area networks. In *Proc. of the IEEE Infocom '94*, Toronto, June 1994.

[27] H. Shen and P. Dewan. Access control for collaborative environments. In *Proc. CSCW'92*, pages 51–58. ACM, November 1992.

[28] P.K. Srimani and S.R. Das. *Distributed Mutual Exclusion Algorithms*. IEEE Computer Society Press, Los Alamitos, CA, 1992.

[29] R. Steinmetz and K. Nahrstedt. *Multimedia: Computing, Communications and Applications*. Prentice Hall, Upper Saddle River, NJ, 1995.

[30] M.B. Walker. Smooth transitions in conversational turn-taking: Implications for theory. *Journal of Psychology*, 110(1):31–37, Jan. 1982.

[31] R. Yavatkar, J. Griffioen, and M. Sudan. A reliable dissemination protocol for interactive collaborative applications. In *Proc. ACM Multimedia*, pages 333–344, San Francisco, Nov. 1995.