INTERNET DRAFT
File: draft-quinn-multicast-apps-00.txt
Expiration: May 1999

B.Quinn
IP Multicast Initiative
November 1998

IP Multicast Applications: Challenges and Solutions

Status of this Memo

This document is an Internet-Draft. Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

To view the entire list of current Internet-Drafts, please check the "1id-abstracts.txt" listing contained in the Internet-Drafts Shadow Directories on ftp.is.co.za (Africa), ftp.nordu.net (Northern Europe), ftp.nis.garr.it (Southern Europe), munnari.oz.au (Pacific Rim), ftp.ietf.org (US East Coast), or ftp.isi.edu (US West Coast).

Abstract

This document highlights the challenges of creating multicast applications, and describes the solutions available or under development. It provides a taxonomy of multicast applications in terms of their requirements, and discusses some existing multicast-based protocols. Many of the solutions--especially in the areas of reliable multicast data delivery, congestion control, and security--have not yet emerged from the research realms. We describe the general state of on-going research in these areas, highlighting the strategies under investigation.

Table of Contents

1. Introduction

IP Multicast will play a prominent role on the Internet in the coming years. It is a requirement, not an option, if the Internet is going to scale. Multicast allows application developers "to add more functionality without significantly impacting the network" [Bradner].

Developing multicast-enabled applications is ostensibly simple. Having datagram access allows any application to send to a multicast address. A multicast application need only increase the Internet Protocol (IP) time-to-live (TTL) value to more than 1 (the default value) to allow outgoing datagrams to traverse routers. To receive a multicast datagram, applications join the multicast group, which transparently generates an IGMP [IGMPV2] group membership report.

This apparent simplicity is deceptive, however. Enabling multicast support in applications and protocols that can scale well on a heterogeneous network is a significant challenge. Specifically, sending constant bit rate datastreams, reliable data delivery, security, and managing many-to-many communications all require special consideration. Some solutions are available, but many of these services are still active research areas.

1.1 Motivation

The purpose of this document is to provide an orientation for application developers to the types of services multicast applications need, and the current state-of-the-art of their development.

Multicast-based applications and services will play an important role in the future of the Internet as continued multicast deployment encourages their use and development. It is important that developers be aware of the issues and solutions available--and especially of their limitations--in order to avoid protocols that negatively impact networks (thereby counter-acting the benefits of multicast) or wasting their efforts "re-inventing the wheel."

The hope is that by raising developers' awareness, we can adjust their expectations of finding solutions and lead them to successful, scalable, and "network-friendly" development efforts.

1.2 Focus

Our initial premise is that the multicast infrastructure is transparent to applications, so it is not directly relevant to this discussion. Our focus here is on multicast application protocol services, so this document explicitly avoids any discussion of multicast address management and routing issues. We identify and describe the multicast-specific issues involved with developing applications.

We assume the reader has a general understanding of the mechanics of multicast, and in this respect we intend to compliment other introductory documents [Maufer]. Since this is an introductory survey rather than a comprehensive examination, we refer readers to other multicast application requirements descriptions [LSMA] for more detail.

In the remainder of this document we first define the term "IP multicast enabled network," the multicast infrastructure. Next we describe the types of new functionality that multicast applications can enable and their requirements. We then examine the services that satisfy these requirements, the challenges they present, and provide a brief survey of the solutions available or under development. We wrap up with a discussion of other application considerations, such as session management and application programming interfaces (APIs).

2. IP Multicast Enabled Network

An "IP multicast-enabled network" provides end-to-end services in the IP network infrastructure to allow any IP host to send datagrams to an IP multicast address that any number of other IP hosts can receive. This requires two essential protocol components:

1) An IP host-based protocol to allow a receiver application to notify a local router(s) that it has joined the group

2) An IP router-based protocol to allow any routers with multicast group members (receivers) on their local networks to communicate with other routers to ensure that all datagrams sent to the group address are forwarded to all receivers

Additionally, a complete IP multicast-enabled network also requires a global address management infrastructure designed to reasonably avoid "address collisions" [MASC]. An address collision occurs when two different applications send to the same multicast address in the same date/time slot for different purposes, thereby possibly "polluting" each other's datastream. An address management infrastructure includes a host-based protocol mechanism to allow an application to request dynamic address allocations for "lease" periods [MDHCP].

At the time of this writing some of these services are not standardized or deployed. Specifically, global address management and intra-domain multicast routing are incomplete. Nonetheless, in the remainder of this document we assume that the multicast-enabled network is already full-service in these respects, and ubiquitous. Although the global Internet is not yet fully multicast-enabled, a large and growing portion is and many enterprise networks (Intranets) are also, so this perspective is relevant today.

3. IP Multicast Application Taxonomy

With an IP multicast-enabled network available, some unique and powerful applications and application services are possible. "Multicast enables coordination - it is well suited to loosely coupled distributed systems (of people, servers, databases, processes, devices...)" [Estrin].

The sender and receiver relationships are primarily what differentiate multicast applications from unicast applications. In this respect, we can characterize three very general categories of multicast applications:

One-to-Many (1toM): A single host sending to two or more (n) receivers

Many-to-One (Mto1): Any number of receivers sending data back to a (source) sender via unicast or multicast

Many-to-Many (MtoM): Any number of hosts sending to the same multicast group address, as well as receiving from it

For each of these multicast application categories, we provide a list of application and protocol examples. These lists are not comprehensive, but include the prominent multicast application types in each category. We reference the items in these lists in the remainder of this document as we describe their specific service requirements, define the challenges they present, and reference solutions available or under development.

In section 3.4 we provide a summary of the bandwidth and delay requirements for the applications listed below.

3.1 One-to-Many Applications

When people think of multicast, they most often think of broadcast-based multimedia applications: television (video) and radio (audio). This is a reasonable analogy and indeed these are significant multicast applications, but these are far from the extent of applications that multicast can enable. Audio/Video distribution represents a fraction of the multicast application possibilities, and most do not have analogs in today's consumer broadcast industry.

a) Scheduled audio/video (a/v) distribution: Lectures, presentations, meetings, or any other type of scheduled event whose multimedia coverage could benefit an audience (i.e. television and radio "broadcasts"). One or more constant-bit-rate (CBR) datastreams and relatively high-bandwidth demands characterize these applications. When more than one datastream is present--as with an audio/video combination--the two are synchronized and one typically has a higher priority than the other(s). For example, in an a/v combination it is more important to ensure a legible audio stream, than perfect video.

b) Push media: News headlines, weather updates, sports scores, or other types of non-essential dynamic information. "Drip-feed," relatively low-bandwidth data characterize these applications.

c) Caching: Web site content, executable binaries, and other file-based updates sent to distributed replication/caching sites

d) Announcements: Network time, multicast session schedules,

random numbers, keys, configuration updates, (scoped) network

locality beacons, or other types of information that are commonly useful. Their bandwidth demands can vary, but generally they are very low bandwidth.

e) Monitoring: Stock prices, Sensor equipment (seismic activity, telemetry, meteorological or oceanic readings), security systems, manufacturing or other types of real-time information. Bandwidth demands vary with sample frequency and resolution, and may be either constant-bit-rate or bursty (if event-driven).

3.2 Many-to-One Applications

Many-to-one applications are typically two-way request/response applications, where either end (the "many" or the "one") may generate the request.

A common challenge among this type of application is dealing with the potential of a "response storm," also known as the "implosion problem." This occurs when receivers overwhelm the sender by forwarding their responses simultaneously. This problem is also common in reliable data delivery and adaptive applications as we describe later along with avoidance strategies.

f) Resource Discovery: Service Location, for example, leverages IP Multicast to enable "anycast" capability: A multicast receiver to send a query to a group address, to elicit responses from the closest host(s) so they can satisfy the request. The responses might also contain information that allows the receiver to determine the most appropriate (e.g. closest) service provider to use.

g) Data Collection: This is the converse of a one-to-many "monitoring" application described earlier. In this case there may be any number of distributed "sensors" that send data to a data collection host. The sensors might send updates in response to a request from the data collector, or send continuously at regular intervals, or send spontaneously when a pre-defined event occurs. Bandwidth demands can vary based on sample frequency and resolution.

h) Auctions: The "auctioneer" starts the bidding by describing whatever it is for sale (product or service or whatever), and receivers send their bids privately or publicly (i.e. to a unicast or multicast address).

i) Polling: The "pollster" sends out a question, and the "pollees" respond with answers.

j) Juke Box: Allows near-on-demand a/v playback. Receivers use an "out-of-band" protocol mechanism (via web, email, unicast or multicast requests, etc.) to send their playback request into a scheduling queue [IMJ].

3.3 Many-to-Many Applications

The many-to-many capabilities of IP multicast enable the most unique and powerful applications. Many-to-many applications are characterized by two-way communications where any host may send to the group as well as receive from it. Since each host may receive data from multiple senders while it also sends data, many-to-many applications often present complex coordination and management challenges.

k) Multimedia Conferencing: Audio/Video and whiteboard comprise the classic conference application. Having multiple datastreams with different priorities characterizes this type of application. Co-ordination issues--such as determining who gets to talk when--complicate their development and usability. There are common heuristics and "rules of play", but no standards exist for managing conference group dynamics.

l) Synchronized Resources: Shared distributed databases of any type (schedules, directories, as well as traditional Information System databases).

m) Concurrent Processing: Distributed parallel processing.

n) Collaboration: Shared document editing.

o) Distance Learning: This is a one-to-many a/v distribution application with "upstream" capability that allows receivers to question the speaker(s).

p) Chat Groups: These are like text-based conferences, but may also provide simulated representations ("avatars") for each "speaker" in simulated environments.

q) Distributed Interactive Simulations [DIS]: Each object in a simulation multicasts descriptive information (e.g. telemetry) so all other objects can render the object, and interact as necessary. The bandwidth demands for these can be tremendous, as the number of objects and the resolution of descriptive information increases.

r) Multi-player Games: Many multi-player games are simply distributed interactive simulations, and may include chat group

capabilities. Bandwidth usage can vary widely, although

today's first-generation multi-player games attempt to minimize bandwidth usage to increase the target audience (many of whom still use dial-up modems).

s) Jam Sessions: Shared encoded audio (e.g. music). The bandwidth demands vary based on the encoding technique, sample rate, sample resolution, number of channels, etc.

3.4 Bandwidth and Delay Requirements Summary

For quick reference, we've plotted the bandwidth and delay characteristics of the multicast applications in our lists. Figure 1 shows multicast applications approximate bandwidth requirements.

We provide this summary here rather than in section 4 (Multicast Service Requirements) because bandwidth and delay requirements are common to unicast as well as multicast network applications. Unicast and multicast applications both need to design applications to adapt to the variability of network conditions. But as we describe in section 4.1, it is the need to accommodate multiple heterogeneous multicast receivers--with their diversity of bandwidth capacity and delivery delays--that presents the unique challenge for multicast applications to satisfy these requirements.

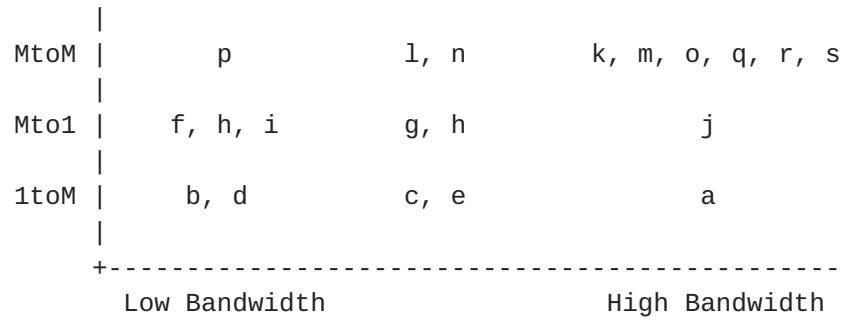

Figure 1: Bandwidth Requirements of applications

Aside from those with time-sensitive data (e.g. stock prices, and real-time monitoring information), most one-to-many applications have a high tolerance for delay and delay variance (jitter). Constant bit-rate (CBR) data--such as streaming media (audio/video)--are sensitive to delivery delay variations (jitter), but applications commonly counteract the effects by buffering data and delaying playback.

Most many-to-one and many-to-many multicast applications are intolerant of delays because they are bidirectional, interactive and request/response dependent. As a result, delays should be minimized, since they can adversely affect the application's

usability.

This need to minimize delays is most evident in (two-way) conference applications, where users cannot converse effectively if the audio or video is delayed more than 500 milliseconds. For this and other examples see Figure 2, which plots multicast applications on a (coarse) scale of sensitivity to delivery delays.

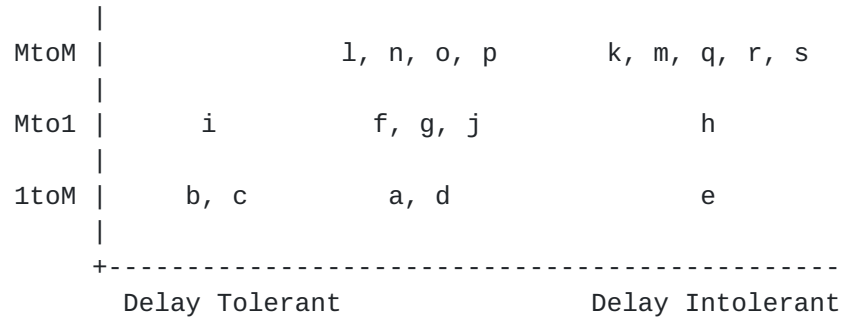

Figure 2: Delay tolerance of application types

For delay-intolerant multicast (or unicast) applications, quality of service (QoS) is the only option. IP networks currently provide only "best effort" delivery, so data are subject to variable router queuing delays and loss due to network congestion (router queue overflows). IP QoS standards do exist now [RSVP] and efforts to enable end-to-end QoS support in the Internet are underway [DiffServ].

However, QoS support is an IP network infrastructure consideration and relevant to unicast as well as multicast. Since our focus is on multicast-specific application services, further discussion of the QoS protocols and services is beyond the scope of this document.

4. Multicast Service Requirements

The application categories described in the previous section are very general in nature. Within each category and even among each of the application types, specific application instances have a variety of application requirements. One-to-many application types are relatively simple to develop, but as we pointed out there are challenges involved with developing many-to-one and many-to-many applications.

The most challenging multicast application service requirements can be summarized into three categories:

Heterogeneous Receivers - Sending to receivers with a wide variety of bandwidth capacities, latency characteristics, and network congestion

Reliable Data Delivery - Ensuring that all data sent is received by all receivers

Security - Ensuring content privacy among dynamic multicast group memberships, and limiting senders

In the remainder of this document, we will describe the challenges involved with enabling each of these application services, and the status of standardizing possible solutions.

4.1 Heterogeneous Receivers

The Internet is a network of networks. IP's strength is its ability to enable seamless interoperability between hosts on disparate network media, the heterogeneous network.

When two hosts communicate via unicast--one-to-one--across an IP network, it is relatively easy for senders to adapt to varying network conditions. The Transmission Control Protocol (TCP) provides reliable data transport, and is the model of "network friendly" adaptability.

TCP receivers send acknowledgements back to the sender for data delivered. A TCP sender detects data loss from the data sent that is not acknowledged. When it detects data loss, TCP infers that there is network congestion or a low-bandwidth link, and adapts by throttling down its send rate [SlowStart].

User Datagram Protocol (UDP) does not enable a receiver feedback loop the way TCP does, since UDP does not provide reliable data delivery service. As a result, it also does not have a loss detection and adaptive congestion control mechanism as TCP does. However, it is possible for a unicast UDP application to enable similar adaptive algorithms to achieve the same result, or even improve on it.

A unicast UDP application that uses a feedback mechanism to detect data loss and adapt the send rate, can do so better than TCP. TCP automatically reduces the "congestion window" when data loss is detected, although the updated send rate may be slower than a CBR audio/video stream requires. When a UDP application detects loss, it can adapt the data itself to accommodate the lower send rate. For example, a UDP application can:

- Reduce the data resolution (e.g. send lower fidelity audio/video by reducing sample frequency or frame rate) to reduce data rate.

- Modify the data encoding to add redundant data (e.g. forward error correction) offset in time to avoid fate sharing. This could also be "layered", so a percentage of data loss will simply reduce fidelity rather than corrupt the data.

- Reduce the send rate of one datastream in order to favor another of higher priority (e.g. sacrifice video in order to ensure audio delivery).

- Send data at a lower rate (i.e. with a different encoding) on a separate multicast address and/or port number for high-loss receivers.

However, with multicast applications--one-to-many or many-to-many--which have multiple receivers, the feedback loop design needs modification. If all receivers return data loss reports simultaneously, the sender is easily overwhelmed in the storm of replies. This is known as the "implosion problem."

Another problem is that heterogeneous receiver capabilities can vary widely due to the wide range of (static) network media bandwidth capabilities and dynamically due to transient traffic conditions. If a sender adapts its send rate and data resolution based on the loss rate of its worst receiver(s), then it can only service the lowest common denominator. Hence, a single "crying baby" can spoil it for all other receivers.

Strategies exist for dealing with these heterogeneous receiver problems. Here are two examples:

Shared Learning - When loss is detected (i.e. a sequenced packet isn't received), a receiver starts a random timer. If it receives a data loss report sent by another receiver as it waits for the timer to expire, it stops the timer and does not send a report. Otherwise, it sends a report when the timer expires. The Real-Time Protocol and its feedback-loop counterpart Real-Time Control Protocol [RTP/RTCP] employ a strategy similar to this to keep feedback traffic to 5 percent or less than the overall session traffic. This technique was originally utilized in IGMP.

Local Recovery - Some receivers may be designated as local distribution points or "transcoders" that either re-send data locally (possibly via unicast) when loss is reported or they re-encode the data for lower bandwidth receivers before re-sending. No standards exist for these strategies, although "local recovery" is used by several reliable multicast protocols.

Adaptive multicast application design for heterogeneous receivers is still an active area of research. The fundamental requirements are to maximize application usability, while accommodating network conditions in a "network friendly" manner. In other words, congestion detection and avoidance are (at least) as important in protocol design as the user experience. The adaptive mechanisms must also be stable, so they do not adapt too quickly--changing encoding and rates based on too little information about what may be a transient condition--to avoid oscillation.

4.2 Reliable Data Delivery

Many of the multicast application examples in our list--like audio/video distribution--have loss-tolerant data content. In other words, the data content itself can remain useful even if some of it is lost. For example, audio might have a short gap or lower fidelity but will remain legible despite some data loss.

Other application examples--like caching and synchronized resources--require reliable data delivery. They deliver content that must be complete, unchanged, in sequence, and without duplicates. The "Loss Intolerant" column in Figure 3 shows a list of applications with this requirement, while the others can tolerate varying levels of data loss. The tolerance levels are typically determined by the nature of the data and the encoding in use.

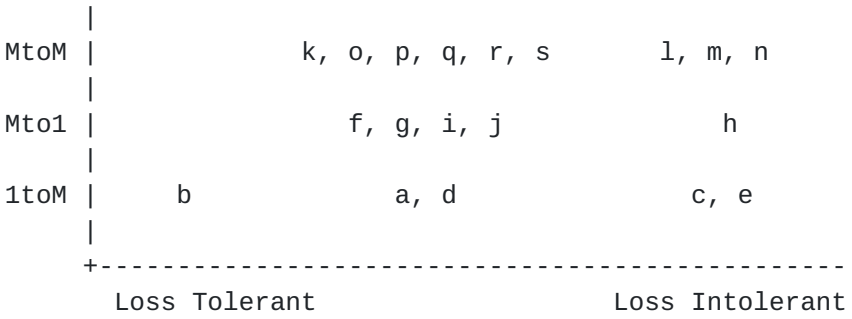

Figure 3: Reliability Requirements of Application types

Some of the challenges involved with enabling reliable multicast transport are the same as those of sending to heterogeneous receivers, and some solutions are similar also. For example, many reliable multicast transport protocols avoid the implosion problem by using negative acknowledgements (NAKs) from receivers to indicate what was lost. They also use "shared learning" whereby receivers listen to others' NAKs and then listen for the resulting retransmission of data, rather than requesting retransmission by sending a NAK themselves.

Although reliable delivery cannot change the data sent--except, perhaps, to use a loss-less data compression algorithm--they can use other adaptive techniques like sending redundant data, or adjusting the send rate.

Although many reliable multicast protocol implementations exist [Obraczka], and a few are already available in commercial products, none of them are standardized. Work is ongoing in the "Reliable Multicast" research group of the Internet Research Task Force [IRTF] to provide a better definition of the problem, the multicast transport requirements, and protocol mechanisms.

Scalability is the paramount concern, and it implies the general need for "network friendly" protocols that detect and avoid congestion as they provide reliable delivery. Other considerations are protocol robustness, support for "late joins", group management and security (which we discuss next).

The current consensus is that due to the wide variety of multicast application requirements--some of which are at odds--no single multicast transport will likely be appropriate for all applications. As a result, most believe that we will eventually standardize a number of reliable multicast protocols, rather than a single one.

4.3 Security

For any IP network application--unicast or multicast--security is necessary because networks comprise users with different levels of trust.

Network application security is challenging, even for unicast. And as the need for security increases--gauged by the risks of being without it--the challenges increase also. Security system complexity and overhead is commensurate with the protection it provides. "No one can guarantee 100% security. But we can work toward 100% risk acceptance ...Strong cryptography can withstand targeted attacks up to a point--the point at which it becomes easier to get the information some other way ...A good design starts with a threat model: what the system is designed to protect, from whom, and for how long." [Schneier]

Multicast applications are no different than unicast applications with respect to their need for security, and they require the same basic security services: user authentication, data integrity, data privacy and user privacy (anonymity). However, enabling security for multicast applications is even more of a challenge than for unicast. Having multiple receivers makes a difference, as does their heterogeneity and the dynamic nature of multicast group

memberships.

Multicast security requirements can include any combination of the following services:

- Limiting Senders - Controlling who can send to group addresses
- Limiting Receivers - Controlling who can receive
- Limiting Access - Controlling who can "read" multicast content
- Verifying Content - Ensuring that data originated from an authenticated sender and was not altered en route
- Protecting Receiver Privacy - Controlling whether sender(s) or other receivers know receiver identity

This list is not comprehensive, but includes the most commonly needed security services. Different multicast applications and different application contexts can have very different needs with respect to these services, and others. "Two main issues emerge, where the performance of current solutions leaves much to be desired" [Canetti]:

- Individual authentication - when, how and to whom are encryption keys distributed?
- Membership revocation - when, why, and how are encryption keys revoked?

Performance is largely a factor when a user joins or leaves a group. For example, methods used to authenticate potential group members during joins or re-keying current members after a member leaves can involve significant processing and protocol overhead and result in significant delays that affect usability.

Like reliable multicast, secure multicast is also still under investigation in the Internet Research Task Force [IRTF]. Protocol mechanisms for many of the most important of these services--such as limiting senders--have not yet been defined, let alone developed and deployed.

As is true for reliable multicast, the current consensus is that no single security protocol will satisfy the wide diversity of sometimes-contradictory requirements among multicast applications. Hence, multicast security will also likely require a number of different protocols.

5. Other Considerations

In the previous section we surveyed the most challenging service requirements of multicast applications. There are a few other more generic requirements that we haven't mentioned yet that deal specifically with creating and managing multicast application instances. Two of them--session management and join/leave latency--are borderline infrastructure services required as part of a multicast-enabled network, but requiring some application interaction. The other--Service API definition--is directly related to application development flexibility and control.

5.1 Session Management

Multicast applications need a "namespace" that provides session directory services that can be used to co-ordinate application schedules and resources, and describe session attributes. These map multicast address and port combinations to a date and time, content description, and other session attributes (e.g. bandwidth and delay requirements, encoding, security and authorization methods, etc.).

The session description protocol [SDP] is designed for this purpose, but it does not provide the transport for dissemination of these session descriptions, nor does it enable the address allocation and management. SDP only provides the syntax for describing session attributes.

SDP session descriptions may be conveyed publicly or privately by means of any number of transports including web (HTTP) and MIME encoded email. The session announcement protocol [SAP] is the de facto standard transport and many multicast-enabled applications currently use it. SAP limits distribution via multicast scoping, but the current protocol definition has scaling issues that need to be addressed. Specifically, the initialization latency for a session directory can be quite long, and it increases in proportion to the number of session announcements. This is largely a multicast infrastructure issue, however, as this level of protocol detail should be transparent to applications.

5.2 Join/Leave Latency

Some applications are sensitive to the latency involved with joining and leaving a group. For example, using distributed a/v as a multicast-based "television" must allow users to "channel surf" as they do now, so any delays changing channels--leaving one group and joining another-- will affect usability. Distributed interactive simulations are sensitive to join/leave latency also, particularly

when trying to represent fast moving objects that may quickly enter

and exit the scope of a simulated environment (e.g. low-flying, fast-moving aircraft).

We have not considered the leave/join latency issue thus far, since applications cannot affect its control. Hence, we consider it a feature of a multicast-enabled network [IGMPv2] and beyond the scope of this document.

5.3 Service APIs

In some cases, the protocol services mentioned in this document can be enabled transparently by passive configuration mechanisms and "middleware." For example, it is conceivable that a UDP implementation could implicitly enable a reliable multicast protocol without the explicit interaction of the application.

Sometimes, however, applications need explicit access to these services for flexibility and control. For example, an adaptive application sending to a heterogeneous group of receivers using RTP may need to process RTCP reports from receivers in order to adapt accordingly (by throttling send rate or changing data encoders, for example) [RTP API]. Hence, there is often a need for service APIs that allow an application to qualify and initiate service requests, and receive event notifications.

Network APIs generally reflect the protocols they support. Their functionality and argument values are a (varying) subset of protocol message types, header fields and values. Although some protocol details and actions may not be exposed in APIs--since many protocol mechanics need not be exposed--others are crucial to efficient and flexible application operation.

A more complete examination of the application services described in this document might also identify the protocol features that could be mapped to define a (generic) API definition for that service. APIs are often controversial, however. Not only are there many language differences, but it is also possible to create different APIs by exposing different levels of detail in trade-offs between flexibility and simplicity.

6. Security Considerations

See section 4.4

7. References

[Bradner] S. Bradner, "Internet Protocol Multicast Problem Statement", <draft-bradner-multicast-problem-00.txt>, September 1997, Work in Progress

[Canetti] R. Canetti, B. Pinkas, "A taxonomy of multicast security issues(temporary version)", <draft-canetti-secure-multicast-taxonomy-00.txt>, May 1998, Work in Progress

[DiffServ] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, K. Nichols, and M. Speer, "A Framework for Use of RSVP with Diff-serv Networks", Internet Draft <draft-ietf-diffserv-rsvp-00.txt>, June 1998

[DIS] J.M.Pullen, M. Mytak, C. Bouwens, "Limitations of Internet Protocol Suite for Distributed Simulation in the Large Multicast Environment", <draft-ietf-lsma-limitations-03.txt>, September 1998, Work in Progress

[Estrin] D. Estrin, "Multicast: Enabler and Challenge", Caltech Earthlink Seminar Series, April 22, 1998

[IGMPv2] B. Fenner, "Internet Group Management Protocol, Version 2", RFC 2236, November 1997

[IMJ] K. Almeroth and M. Ammar, "The Interactive Multimedia Jukebox (IMJ):A New Paradigm for the On-Demand Delivery of Audio/Video", Proceedings of the Seventh International World Wide Web Conference, Brisbane, AUSTRALIA, April 1998

[IRTF] A Weinrib, J. Postel, "The IRTF Guidelines and Procedures", RFC 2014, January 1996

[LSMA] P. Bagnall, R. Briscoe, A. Poppitt, "Taxonomy of Communication Requirements, for Large-scale Multicast Applications," <draft-ietf-lsma-requirements-02.txt>, November 1998, Work in Progress

[MASC] D. Estrin, R. Govindan, M. Handley, S. Kumar, P. Radoslavov, D. Thaler, "The Multicast Address-Set Claim (MASC) Protocol", <draft-ietf-malloc-masc-01.txt>, August 1998, Work in Progress

[Maufer] T. Maufer, C. Semeria, "Introduction to IP Multicast Routing," <draft-ietf-mboned-intro-multicast-03.txt>, July 1997, Work in Progress

[MDHCP] B. V. Patel, M. Shah, S. R. Hanna, " Multicast address

allocation based on the Dynamic Host Configuration protocol", <draft-ietf-malloc-mdhcp-01.txt>, August 1998, Work in Progress

[Obraczka] K. Obraczka "Multicast Transport Mechanisms: A Survey and Taxonomy", IEEE Communications Magazine, Vol. 36 No. 1, January 1998

[RM] A. Mankin, A. Romanow, S. Bradner, V. Paxson, "IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols", RFC 2357, June 1998

[RSVP] J. Wroclawski, "The Use of RSVP with IETF Integrated Services", RFC 2210, September 1997

[RTP API] J. Rosenberg, "Columbia RTP Library API Specification," (Note: Does not include RTCP processing), February 1997

[RTP/RTCP] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 1889, January 1996

[SAP] M. Handley, "SAP: Session Announcement Protocol", <draft-ietf-mmusic-sap-00.txt>, November 1996, Work in Progress

[SDP] M. Handley, V. Jacobson, "SDP: Session Description Protocol", RFC 2327, April 1998

[Schneier] B. Schneier, _ Why Cryptography Is Harder Than It Looks", December 1996, http://www.counterpane.com/whycrypto.html

[SlowStart] W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms", RFC 2001, January 1997

8. Author's Address

Bob Quinn
IP Multicast Initiative (IPMI)
Stardust Forums, Inc.
1901 S. Bascom Ave. #333
Campbell, CA 95008

+1 408 879 8080
rcq@ipmulticast.com