



Helix Mobile Server Rate Control for Mobile Networks

Version 1.8

This document contains trade secrets and proprietary information belonging to RealNetworks, Inc. No use or disclosure of the information contained herein is permitted without the prior written consent of RealNetworks, Inc.

© 2008 RealNetworks, Inc.
All Rights reserved.

Helix, the Helix logo, RealAudio, RealMedia, RealNetworks, RealOne, RealSystem, RealVideo, the Real Bubble logo, and SureStream are trademarks or registered trademarks of RealNetworks, Inc. in the United States of America and other countries.

All other trade names, trademarks, or registered trademarks are trade names, trademarks or registered trademarks of their respective companies.



2601 Elliott Avenue
Seattle, Washington 98121
Phone: (206) 674.2700
Fax: (206) 674.2699
www.realnetworks.com



Table of Contents

- 1 Overview 6
 - 1.1 Purpose..... 6
 - 1.2 Abbreviations..... 6
- 2 Theory of Operation..... 7
- 3 Technical Components 9
 - 3.1 Overview..... 9
 - 3.2 Rate Control 9
 - 3.3 Rate Adaptation 11
- 4 Requirements 14
 - 4.1 Client Requirements..... 14
 - 4.2 Server Requirements 15
 - 4.3 Content Requirements..... 16
 - 4.3.1 Encoding Ranges 16
 - 4.3.2 RealMedia SureStream Content..... 16
 - 4.3.3 3GPP Multi-Rate Content (.mrc) 17
- 5 Procedures for enabling a New Terminal 18
 - 5.1 Configuration 18
 - 5.1.1 Overview..... 18
 - 5.1.2 Client Specific Parameters..... 18
 - 5.1.3 User Agent 18
 - 5.1.4 Client Technology 19
 - 5.1.5 RTCP RR Algorithm 19
 - 5.1.6 Preroll Calculation..... 20
 - 5.1.7 Buffer Model 20
 - 5.1.8 Rebuffer Criteria..... 20
 - 5.1.9 Network Support 21
 - 5.2 Creating a Baseline Configuration File 21
 - 5.2.1 Overview..... 21
 - 5.2.2 Procedure..... 23
 - 5.3 Validating the Configuration 27
 - 5.3.1 Run Time Tracing for Rate Control..... 27



TABLE OF CONTENTS

5.3.2	Run Time Tracing for Buffer Control and Stream Switching	30
6	Content encoding Guidelines.....	33
6.1	Recommendations	33
6.2	Considerations	34
7	Performing a Test.....	35
7.1	What to collect from a performed test.....	35
8	Appendix A – Variable Definitions	36
9	Appendix B - Sample RDF FILE.....	40
10	Appendix C – Sample MDP config	41
11	Appendix D – Data required from Client manufacturer	42
12	References	42
13	Document Control	43
13.1	Contributions	43
13.2	Version Control	43

1 OVERVIEW

1.1 Purpose

The purpose of this document is to provide an overview of the qualifications for the Rate Control feature of the Helix Mobile Server. This document is designed to provide a detailed description of the theory of operation for Rate Control, present a framework for understanding the interdependent technical components that make up this feature and provide technical details for the qualification of this feature for new mobile terminals.

The goal of presenting this information is to provide a baseline understanding of the Rate Control feature from which performance can accurately be evaluated and problems can effectively be addressed, as well as to clarify the requirements of all components (RealNetworks and 3rd party) that contribute to the operation of the Rate Control feature.

1.2 Abbreviations

3GPP	3 rd Generation Partnership Project
3GPP2	3 rd Generation Partnership Project 2
AAC	Advanced Audio Coding
AMR	Adaptive Multi-Rate
BCC	Binomial Congestion Control
CDMA	Code Division Multiple Access
EDGE	Enhanced Data Rates for Global Evolution
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
MDP	Media Delivery Pipeline
MIDI	Musical Instrument Digital Interface
MMS	Multimedia Messaging Service
RDT	Reliable Data Transport
RTCP	Real Time Control Protocol
RTP	Real-time Transport Protocol
RTT	Roundtrip Time
RTSP	Real Time Streaming Protocol
SDP	Session Description Protocol
SMIL	Synchronized Multimedia Integration Language
HTTP	Hypertext Transfer Protocol
PSS	Packet-switched Streaming Service
QoS	Quality of Service
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
URL	Uniform Resource Locator
WCDMA	Wideband Code Division Multiple Access

2 THEORY OF OPERATION

Rate Control is the process of changing the attributes of the media delivered from the server to the client in an effort to optimize the users' experience of the media within the constraints of the underlying network. The purpose of Rate Control on mobile networks is to ensure a continuous experience of the highest fidelity possible, given the narrow bandwidth and varying conditions of the network.

Optimizing the users' experience requires that media with maximal visual fidelity be presented to the user with maximal continuity. Visual fidelity is affected by two parameters, the encoding quality and the magnitude of loss. Encoding quality trades compression for visual fidelity. Loss compromises the ability of the codec to decode frames for presentation to the user. Continuity is determined by the relationship of the channel rate to the media rate. If the channel rate is too low for the media rate, then the user will experience breaks in the continuity of the stream as the client rebuffers data for the codec.

Discontinuities in transmitted media occur because the network is not able to deliver media at the rate required to decode the media in real time. The network rate is stochastically affected by network handover events, congestion, and physical layer (RAN) interference, which cannot be reliably predicted or avoided. Rather, the effects of network rate-changes on the media itself are observed, and actions can be taken based on these observations.

Through the course of operation, the Rate Control components attempt to minimize loss, minimize rebuffering and maximize the encoding rate delivered. Performing Rate Control minimizes loss. Rebuffering is minimized by adapting the media encoding rate to better fit the channel rate of the underlying network. Conversely the encoding rate can be maximized by insuring that the server transmits the highest quality-encoding rate that is best fit to the channel.

The most efficient estimator of the extent to which the network rate is meeting the requirements of the media is a simple queue. New packets arrive on the queue from the network at a dynamic rate, and the packets are removed from the queue at a fixed rate. If packets arrive faster than they are removed, the queue increases in length, if packets arrive slower than they are removed then the queue decreases in length. If the queue is ever emptied before the media presentation completes, the user experiences a discontinuity in the presentation (rebuffering). If the queue is ever filled to its maximal limit, then the encode rate is not optimal for the capabilities of the network.

To prevent a discontinuity in the media presentation either the arrival rate of packets must increase, or the removal rate must decrease. There is a cost associated with each corrective action. If the removal rate is decreased, then the encode rate must decrease and the user will experience a decrease in the quality of the media. If the arrival rate is increased, then the network may be over loaded and packet loss may occur, which also results in compromised quality of the media.

To ensure optimal use of the network resources in order to provide the highest quality presentation to the user, the encode rate may be increased when the queue fills to its maximal limit. Rate Control must therefore find a careful balance between optimizing the encode quality of the media, and minimizing the effects of network packet loss. Further, this balance must be maintained as the dynamics of the network change due to a number of factors.

The dynamics of the network are inferred by measuring Round Trip Time (RTT) and loss, as reported by the client. In order for the server to determine the maximal capacity of the network it must send enough data to either dramatically increase the RTT or induce a loss event. The server attempts to prevent loss events by first responding to large increases in RTT. Loss events typically indicate that the carrying capacity of the network has been reached. The server responds to loss events by decreasing the rate at which it is sending. The upper bound of the link may change, so the server must periodically recheck the upper bounds. This periodic rechecking results in small oscillations in delivery rate over time. Although the server attempts to avoid loss events due to channel optimization, loss events may still occur.



All server-side Rate Control is based on feedback derived at the client, and transmitted to the server. RTCP, specified in IETF RFC 1889 [10], is used to transmit client reception statistics to the server using the RTCP Receiver Report (RR) format.

Clients report the RTT and loss attributes to the server. The interval at which these parameters are reported varies based on the client implementation. The report interval is important because it determines the statistical accuracy of the network measurements over time. The server may not be able to react rapidly to changes in network conditions if the client does not send a report at an appropriately frequent interval to guarantee rapid response at the server.

The server is therefore dependent on statistics reported from the client and the frequency of these reports in order to process this feedback to deduce fundamental metrics on which to base adaptation decisions. Details on specific client requirements and the impact of non-compliance are described in the Requirements section below.

It is not advantageous to change the rate in response to every change in the network rate. Changes in the network rate may happen on very small time scales. Clients may be able to continue to decode the media at an optimal rate, independent of short-term changes in the network rate. Further, frequent and rapid changes in the media rate create a poor experience for the user. Rate changes carry the penalty of a potential for visual distortion during the rate change, although this distortion is not deterministic and is dependant on the client codec implementation.

Adaptation of the media encoding is therefore dependent on the state of the queue referred to earlier and not the inferred network rate. The server maintains a model of this queue based on the reports received from the client, while the queue itself exists at the client and in the network. Media encodings are switched in response to two events. First, if the server's estimation of the queue depth falls to a certain level, the media encoding rate is decreased. Second, if the server's estimation of the queue depth increases to a certain level, then the media encoding rate is increased. This principle has implications for the timing of Rate Control events.

The timing of a switch to a lower encoding rate is dependent on the queue depth prior to a network event that requires a decrease in the encoding rate. If the queue depth is high, then some time will pass after the network changes rates before the server will respond by changing the media rate. The queue must drain to a sufficiently low level before a rate change is required. This principle is followed to allow the client to play through changes in the network rate that occur on short time scales.

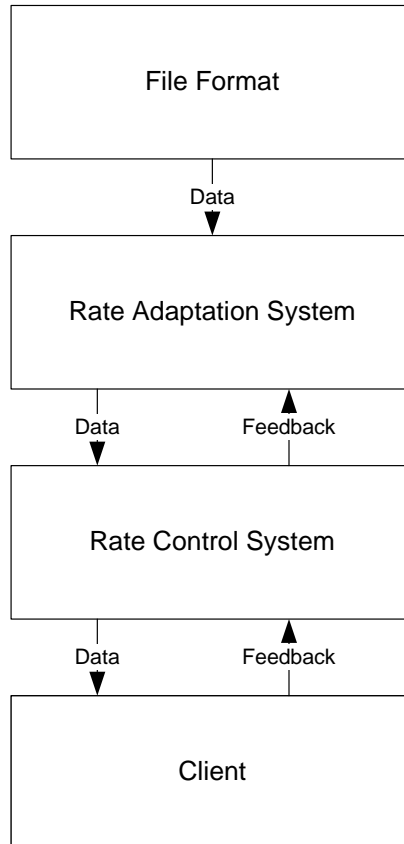
The result of this is a lag in the perceived rate switch point from the network event. The timing of a switch to a higher encoding rate is dependent on the queue depth and on the difference in magnitude between the available encoding rates. This makes the switch point dependant on the encoding ranges selected during the encoding process.

The media encoding scheme also determines the response time for an increase in rate. If the difference in magnitude between rates is large and the queue depth is low prior to the network event, then the encoding change will take longer than if the difference in magnitude between rates were smaller. The visual impact of a rate shift is usually small, but occasionally visual artifacts can appear in the execution of a shift.

3 TECHNICAL COMPONENTS

3.1 Overview

The data flow path for the server-side Rate Control feature is represented below:



The server-side Rate Control feature functions via a combination of Rate Control (modulating the rate at which media is sent over the network) and Rate Adaptation (adjusting the quality of the media to account for changing network conditions). These components are described below.

3.2 Rate Control

The Rate Control subsystem is responsible for modulating the media transmission rate with a specific goal of sending data at the highest possible rate while not over utilizing the capacity of the underlying network. The ancillary effect of not over utilizing the network is a reduction in loss and a dynamic adaptation to changes in network capacity. If network capacity decreases, Rate Control reduces the sending rate. If network capacity increases, Rate Control increases the sending rate accordingly.

Rate Control is concerned primarily with UDP traffic, since TCP employs its own form of congestion control. An application layer framing protocol, RTP or RDT, is assumed to provide network feedback to Rate Control.

Rate Control as employed in the Helix Mobile Server requires feedback on the state of the network in order to perform its Rate Adaptations. The required feedback metrics are the packet loss rate, the round trip time (RTT), packet inter-arrival jitter and the received rate of the delivered stream.

This information is presented to the Rate Control module by the underlying application layer framing protocol. The method for delivering this feedback information is specific to each protocol type.

The Rate Control module's efficiency and efficacy are determined by the frequency of the feedback reports from the media client. Hence, it is imperative that the media client sends the feedback reports periodically. The optimum period is equal to 1 RTT.

For RTP, the RTCP Receiver Report (RTCP RR) is used by the server to deduce the RTT, loss rate, jitter, and received rate. The RTT is calculated according to the algorithm presented in section 6.3.1 of RFC 1889 [10]. The loss rate is calculated according to the algorithm presented in section 6.3.4 of RFC 1889 [10]. The received rate is calculated by using the difference in highest reported sequence numbers between consecutive receiver reports scaled by the size of the packets for that stream, averaged over the time elapsed between the consecutive receiver reports, and adjusted to account for loss.

For RDT, two protocol mechanisms are used to determine the RTT, loss rate, and received rate. RTT is determined by attaching a 4-octet transmission time stamp at the head of each packet. The client echoes the timestamp to the server upon receipt of the packet. When the server receives a timestamp echo, it samples the RTT as the difference between the receipt time of the timestamp echo, and the transmission timestamp contained in the timestamp echo packet. This method is resilient to loss of any single packet or groups of packets and does not require maintaining a sequence space for echo packets. It also introduces minimal protocol overhead that is easily scaled to meet carrier requirements. Loss and received rate are determined through cumulative acknowledgment packets that are sent at 1-second intervals from the client to the server. The ACK packets contain a payload that is a vector of sequence numbers acknowledged. The client explicitly signals loss in vector of sequence numbers not received that is also contained in the ACK packet. Loss is determined through this negative acknowledgment vector, and the received rate is deduced by averaging the number of packets received over the interval between ACK packets.

The application layer framing transport then presents the RTT, loss, and received rate data to Rate Control for processing. A UDP congestion control algorithm, either TFRC or BCC, uses this information to modulate the send rate.

The Helix Server Rate Control module utilizes a variant of the TFRC algorithm presented in [13]. The Rate Control equation presented in [13] remains unchanged. The manner in which loss and RTT are statistically filtered, however, is modified. The averaging mechanism for loss and RTT employs an Infinite Impulse Response (IIR) filter to allow for more rapid and tunable response of the TFRC algorithm to changes in loss and RTT than the TCP-like RTT filter and loss history mechanism presented in RFC 3448 [13]. This responsiveness improves the ability of the congestion control algorithm to recover from service gaps.



One of the goals of TFRC is to insure that the link is optimally used. TFRC's definition of optimality includes the requirement that buffers along the network path are filled to capacity. For this reason TFRC can allow the rate to increase over time, particularly during slow start, in spite of increasing delays as a result of networks buffers being filled. This poses a problem for some mobile networks which may contain network buffers along the link path that exceed 1MB in size. On such networks TFRC may cause large RTT to be evident. Large RTT negatively impacts both the TFRC throughput and the throughput of other flows over the same link path.

To remedy this problem, as of version 10.05 or later of the Helix Mobile Server, the operator can chose to use an alternative Rate Control protocol called Binomial Congestion Control or BCC. BCC is described in detail in an academic paper referenced as [14]. BCC is essentially an abstraction of nonlinear congestion control algorithms that can be parameterized. This allows the Rate Control module to modify the behavior of the congestion control algorithm to be optimized for different behaviors. In this case the BCC implementation can be modified to insure that correct Rate Control response is maintained while not causing the RTT to increase to excessive levels.

The BCC algorithm implemented in the Helix Server is a variant of the algorithm described in [14]. [14] suggests that the BCC rate modulation functions are applied in response to the presence or absence of loss events. The Helix Server implementation instead responds to changes in RTT. An increase in RTT beyond a configurable threshold will cause the BCC decrease function to be used, while a decrease in RTT beyond a configurable threshold will cause the BCC increase function to be used. To understand more about the BCC increase and decrease functions refer to [14].

In the absence of network metric feedback from the application layer framing protocol layer, Rate Control is unable to provide adequate information to TFRC or BCC in order to do optimal Rate Control. Thus, Rate Control will abandon TFRC/BCC and send at a steady rate; this is either the current encoding rate or a lower, safer encoding rate.

The advantages of using Rate Control over a best effort network are clear. The channel has a finite capacity. The availability of this capacity changes over time as background traffic fluctuates. As capacity is reached, the network effectively runs out of memory, dropping new data that is injected to the network. Insuring that a transmitted session does not induce such memory overflows and insuring that the session occupies its share of the channel capacity requires a dynamic adaptive algorithm.

Rate Control has an additional advantage of responding to gaps in service introduced by cell re-selection in wireless networks. Cell re-selection appears to the Rate Control module as a rapid increase of RTT followed by a large loss of data, in the case of hard handover. The increase in RTT causes the Rate Control scheme to limit the sending rate of the media, thereby reducing the amount of data stored in network buffers that will be lost during the handover. Further, once a new cell has been selected, the Rate Control algorithm readily gauges the channel rate of the new cell, allowing the rest of the delivery system to adapt to the new rate,.

By comparison, a non-Rate Controlled stream will experience more loss during hard handover. Sending at the media encode rate regardless of network feedback during a re-selection or congestion event will increase the amount of data that will be or can be lost on the network. In this regard Rate Control can be regarded as a loss mitigation mechanism, thereby improving the user experience.

3.3 Rate Adaptation

The Rate Adaptation subsystem provides scheduling input for the packet flow based on the throughput reports from the Rate Control module, the “3GPP-Link-Char” headers for 3GPP Release 6 compatible clients, and SetDeliveryBandwidth for RTSP/RDT. Rate Adaptation maintains a model of the media client’s buffer as specified in Annex G of 3GPP TS 26.234 [3]. It needs to make sure that each packet sent to the media client does not violate the client buffer constraints. Rate Adaptation uses the initial pre-decoder buffering period, the size of the hypothetical pre-decoder buffer, the peak decoding byte rate, and the decoding macro-block rate to control the buffer state of the media client. The initial pre-decoder buffering period and the decoding byte rate are available through the Capability Exchange profile of the media client. In the event that the Capability Exchange profile is not present, administrators shall be able to configure these values based on the type of media client. In cases where the Capability Exchange information as well as configuration information for the media clients is absent then default values are employed. Having this information available is crucial to a quality playback experience.

3.3.1 Basic Rate Management

The basic theory is as follows: Media clients will have different buffer characteristics and Rate Adaptation has to ensure that it handles each media client appropriately. Administrators will be given the provision to configure preset upshift and downshift depths in a media client’s buffer at which shifts occur. The downshift depth can be defined as the lower threshold of the client’s buffer, and aids in determining when a shift to a lower encoding rate is necessary to sustain a seamless experience at the media client. The upshift depth can be defined as the upper threshold of the client’s buffer, and aids in determining when a shift to a higher encoding rate may be possible to provide a better quality experience to the client. If a shift to a higher encoding rate is not possible Rate Adaptation will slow down the transmission rate to prevent the client’s buffer from overflowing.

Rate Adaptation maintains a model of the client buffer at the server. It first determines the size of the buffer model from the VideoPreDecodeBufferSize value given in the client’s capability exchange RDF profile. When the server starts to send data packets out to the client, each packet sent is at the same time “queued” in the buffer model. Thus, as packets are sent to the client and stored in the client buffer, Rate Adaptation does the same with the buffer model.

The server knows how much data (in bytes) are (virtually) stored in the buffer model, and it knows the media rate of the stored data, thus it can calculate how deep the buffer is filled, in milliseconds. For example, if the media rate is 80 kbps, and the buffer is filled with 20000 bytes of data, this computes to a buffer depth of 2000 milliseconds.

At the beginning of the stream, the server only queues packets into the buffer, corresponding to the client pre-buffering data before it starts playback. When the buffer depth reaches the Preroll value, the client will start playback and, correspondingly, Rate Adaptation will evict packets from the buffer model. The Preroll value is encoded in the content or can be configured in the rmserver.cfg file for a given client in case clients ignore the Preroll value that comes with the content. Some clients may also use a different but similar value, called target time, to determine when to begin playback.

Rate Adaptation then queues a packet into the buffer model when it is sent out, and evicts a packet from the buffer when its playback time is reached. By doing this, Rate Adaptation keeps the buffer model in synch with the client buffer.

The rate at which packets are evicted depends on the media rate of the packets. The rate at which packets are queued depends on the actual delivery rate calculated by the Rate Control component. Thus, if Rate Control

calculates a delivery rate higher than the media rate, the buffer model will fill up, and if Rate Control computes a delivery rate below the media rate, the buffer model will drain.

3.3.1.1 *Upshifts*

If the buffer model's fill level exceeds the configurable upshift depth, Rate Adaptation will issue an upshift request. Whether this request will be fulfilled depends on the other two components. Rate Adaptation determines whether there is a higher media rate available in the multi-rate content file. If there isn't, the request cannot be fulfilled. The media rate remains unchanged. In case a higher media rate is available in the content file, Rate Control determines whether its current calculated delivery rate equals or exceeds the next higher media rate. If so, the upshift will be granted, and the server will now send data of a higher media rate. Otherwise, the request is ignored.

After an upshift has been performed, the rate of evicted packets increases, because packets are evicted from the buffer at their media rate, which has increased after the upshift. Since the queuing rate is still determined by Rate Control based on the network conditions, it is likely that the buffer depth does not increase further and will fall below the upshift depth. However, if the network capacity is high enough that Rate Control can still send at a higher rate than the new media rate, the buffer will fill again and the upshift depth will again be reached, causing the next upshift.

In cases where the network capacity exceeds the highest media rate available in the multi-rate content file, Rate Control (which knows nothing about available media rates) will still cause the server to over send data, which will fill buffer model above the upshift depth and until the configured buffer size (VideoPreDecodeBufferSize) is reached. This means that the client's buffer is now filled to its capacity. Continuing to over send data would result in packet loss and degraded quality of experience. Thus, on reaching the buffer model size, Rate Adaptation blocks the sending of each packet until room is made in the buffer model due to ongoing eviction (rendering and playback) of packets. This effectively reduces the server's delivery rate to the current and highest, media rate of the content file.

In practice, it is often observed that the Rate Adaptation buffer model is filled to the top, while it logs many "Stream blocked" messages into its trace file, and the server effectively sends out data at the highest media rate available in the clip. This stable state indicates that there is more network capacity available than is needed to stream the best quality the clip offers. In this state, the client buffer is also filled to the top and is best possibly prepared to overcome congestion, handovers or other distortions in the network.

3.3.1.2 *Downshifts*

If Rate Control reduces the delivery rate below the current media rate, Rate Adaptation's buffer model will drain. If it reaches the downshift depth, Rate Adaptation will request a downshift. This request will only be fulfilled if a lower media rate is available in a multi-rate content file. After a downshift, the buffer will drain more slowly or stop draining because the eviction rate will decrease to the new media rate.

The goal of a downshift is to prevent the buffer from draining to the bottom, because if that happens, it corresponds to the client buffer being empty and the client rebuffering. Rate Adaptation thus prevents rebuffering at the cost of sending out data in lower quality.

However if network conditions are so bad that Rate Control has to reduce the delivery rate even below the lowest available media rate, the buffer will eventually drain to zero. In this case, Rate Adaptation will assume the client rebuffers, and will re-fill the buffer model by queuing packets without evicting them. Evicting starts again if the abovementioned Preroll value is reached.

In practice, frequent rebuffering is very well reflected in the Rate Adaptation trace files which show frequent draining of the buffer until the buffer depth is zero, and switching into the predecode phase, followed by packets only being queued and not evicted. When the buffer is frequently draining and re-filled, this indicates the network capacity is too low to even stream at the lowest possible quality.

4 REQUIREMENTS

4.1 Client Requirements

Server-side Rate Control of the Helix Server supports a wide range of clients. Any 3GPP Release 5 PSS compliant client implementing the Annex G video buffer model is supported. Further, some clients that support additional methods for adaptation signaling are also supported. As new clients emerge, the Rate Control feature can be readily configured by RealNetworks to support those clients as well.

3GPP Release 5 compliant clients take a generally passive role in Rate Control. The only explicit signaling that occurs between the client and server for Rate Adaptation occurs through RTCP Receiver Reports. The remainder of the adaptation processing occurs and is configured on the server.

The performance of server-side Rate Control is therefore determined in part by the frequency of reports from the client. The frequency of client reports determines the frequency with which the server-side model of the throughput may change. Higher granularity of client feedback improves performance as it allows for higher statistical confidence in the server-side throughput and buffer models. A client that will perform optimally with our system is one that can provide relatively fine-grained reporting to the server.

3GPP Release 5 defines a mechanism by which the server can indicate the rate at which RTCP Receiver Reports are sent. Most released handsets do not conform to this requirement. Stream adaptation performed on handsets of RR frequencies over 1 second is conservative as there is a tendency for throughput estimates to be inaccurate.

Some 3GPP PSS compliant clients report feedback to the server via the RTCP signaling mechanism with a frequency specified in RFC 1889. This frequency ranges in practice from 3-5 seconds, and represents the worse case frequency for the Helix Mobile Server Rate Control feature. The desired RR frequency for optimal server behavior is once per round trip time.



While the Helix Mobile Server Rate Control feature does handle this case with robustness, failure of the client to conform to these minimal requirements may result in regular loss seemingly unrelated to network congestion, periodic freezes of the video frame, or failure of the session altogether (rare).

As defined in 3GPP Release 5 TS 26.234 section 5.3.3.1, clients must conform to the requirements for interpreting the b=RR SDP field presented in SDP.



If the client does not meet these requirements, the client may experience periodic visual or aural distortion that results from congestion-induced loss. The client may also incur rebuffer events that result from the server not being able to readily adapt to changing network conditions due to a lack of immediate feedback.



The server will not be able to make an accurate assessment of rapid changes in bandwidth with a lack of frequent RTCP RRs. The server will not be able to use an accurate measure of the clients video pre decode buffer, potentially resulting in client buffer overruns or underruns.

The Helix Mobile Server Rate Adaptation component manages the state of the client buffer as prescribed in Annex G of 3GPP TS 26.234 [3].

PreDecoderBufferSize and VideoDecodingByteRate are two client-specific parameters that are important for the Rate Control feature to function properly. These two parameters are part of the streaming component in the Capability Exchange profile of the client.

The Rate Control feature was initially implemented with the assumption that mobile clients would send an x-wap-profile-header, which contains a URI to the CC/PP profile of the client in the RTSP conversation. HUSM would then fetch the client's profile via http to obtain the values of PreDecoderBufferSize and VideoDecodingByteRate.



Some clients do not present an URI to an RDF file, and some specify the RDF file, but it does not contain values for all of the PSS Vocabulary, particularly PreDecoderBufferSize and VideoDecodingByteRate. These observations have been made from empirical evidence during testing of the Rate Control feature.

Lack of these variables also implies lack of conformance with 3GPP 26.234 Annex G which is required for Rate Control since it defines a model that can be assumed by the client and server for modeling the client pre decode buffer.

For this reason, local client profiles known as Resource Document Framework (.rdf) files were created and installed with the Helix Mobile Server under the ClientProfiles directory for each client. For most clients where data was unavailable, the PreDecoderBufferSize and VideoDecodingByteRate are set to default values as prescribed by 3GPP TS 26.234 [3].

The implication of using default values is that, while these values may work in some cases, they will not work in all cases. Lack of accurate values for these variables may result in inadvertent overruns or underruns of the client buffer resulting in data loss or rebuffering events, respectively.

4.2 Server Requirements

The Helix Mobile Server performs Rate Control in response to network state information signaled by the client via RTCP. Accordingly, Rate Control and client buffer modeling are functions of the server. The server Rate Control and adaptation components are highly configurable. The configuration is segmented to be specific to individual clients, identified by their respective User-Agent strings.

When a client requests content from the server with RTSP, the User-Agent string is presented to the server in an RTSP header. The User-Agent string is used to select a configuration from a set of Rate Control configuration blocks. If there is no match for the specified User-Agent, then a default setting is used. The default configuration can be specified as well. In this manner, the Rate Control feature can be configured to exhibit different behaviors for individual clients. The purpose of this semantic is to accommodate the diversity of streaming media clients that are presently in deployment.

4.3 Content Requirements

4.3.1 Encoding Ranges

The selection of the encoding rates is critical to the performance of Rate Adaptation. There are two factors that contribute to the selection of rates.

1. The first is the magnitude of the rates themselves. The selected rates must be appropriate for the channel over which the media will be streamed. This means that the encoded rate plus the protocol overhead must be less than the advertised bearer rates for the target network.

For example, assuming 8 kbps per slot on GPRS networks, the raw GPRS throughput for each slot allocation will be:

- 1 – 8 kbps
- 2 - 16 kbps
- 3 - 24 kbps
- 4 - 32 kbps

The lowest aggregate rate that can be encoded by the Helix Mobile Producer is 14 kbps.

The RTP/UDP/IP overhead associated with low packet sizes is as much as 5 kbps. There is also a variable amount of overhead associated with Rate Control. Therefore, for 1-3 GPRS Timeslots only the minimal encoding rate of 14 kbps is reasonable. For 4 timeslots a rate of 24 kbps is reasonable.

2. The second factor is the difference in magnitude between rates. If the difference is large, the server must make large steps in throughput utilization to adapt the delivered media to higher rates. The result is that failed attempts to increasing the transmission are potentially expensive in terms of loss perceived by the end user.

4.3.2 RealMedia SureStream Content

RealAudio and RealVideo clips can use SureStream technology to encode multiple streams at different bandwidths in a single clip. When a RealPlayer requests a clip Helix Mobile Server delivers the stream best suited for that player's connection speed. The Helix Mobile Server and the RealPlayer can switch between streams to compensate for changing network conditions.



4.3.3 3GPP Multi-Rate Content

The supported video and audio codecs allowed in the input file are provided in the table below:

File Format	Supported Video Codecs	Supported Audio Codecs
3GPP	MPEG-4 SVP, H.263 Profile 0, H.264	AAC, AAC+, AMR-NB, AMR-WB

Any bandwidth 3GPP files are supported but for MPEG4 only files from similar video bitrate bandwidth ranges known as Profile Levels can be combined. The following table defines the codecs supported for each of the input file formats supported:

Available Profile Levels supported are:

Profile Level 0: GPRS at DL: 20kbps (GPRS) to UMTS Streaming RAB at DL: 64kbps (UMTS)

Profile Level 1: UMTS Interactive RAB DL:64kbps to DL:128kbps

Profile Level 2: UMTS Interactive RAB DL:128kbps to DL:384kbps

For H.263, files from any Profile Level may be combined.

Any one file may contain one each of the above audio and video codecs. Any other file types not listed above will not be interoperable with Helix Server.

The output file may contain any number streams; each with a different video bitrate and/or video codec. 3GPP Release 6 allows for groupings of distinct streams based on criteria such as codec or language. For MPEG4 streams, the bitrate for each stream must be within a single Profile Level as described above.

Many streaming media formats, including QuickTime and MPEG-4, contain a hint track that provides Helix Mobile Server with information about how to stream the clip's media packets. Although you can typically encode these clips without a hint track, doing so is recommended only for downloaded media rather than streamed media. If the format can include a hint track, add the track when encoding the clip to ensure that the clip streams well.

Helix Mobile Server generally can stream MPEG-4 or 3GPP clips encoded with any codec, as long as the clips include a hint track. In addition, Helix Mobile Server can stream clips encoded with certain codecs whether or not the clips contain hint tracks. The following are the MPEG-4 and 3GPP audio and video codecs that do not need to include hint tracks. For these codecs, Helix Mobile Server refers to the hint track if it is present, but still streams the media packets if the track is absent:

- H.263 profile 0 and profile 3
- MPEG-4 Part 2 simple video
- AAC low complexity
- AMR-NB (narrowband) and AMR-WB (wideband)

Reference the Helix Mobile Server Administration Guide for details on supported codecs, their mime type and packetization format for each file format.

5 PROCEDURES FOR ENABLING A NEW TERMINAL

5.1 Configuration

5.1.1 Overview

This section describes the procedure used to configure Rate Adaptation support for a given client and network. The goal of the configuration process is to determine correct values for the set of parameters that define the bounds of Rate Adaptation operation. Correctly identifying and configuring these bounds allows the Rate Adaptation feature to operate as effectively as possible.

The Rate Adaptation feature is highly flexible. This flexibility is required to support a broad set of clients that vary in their feature-completeness, 3GPP specification compliance and their implementation details. It is not valid to assume that all clients are either implemented correctly or that their standards compliance is synonymous with a given set of implementation assumptions.

The Rate Adaptation feature gives the server operator the ability to support an advanced functionality to a widely heterogeneous client population. With this ability, however, comes a particular responsibility to accurately identify client-specific variables that impact the functionality of Rate Adaptation. Failure to do so is the most common pitfall of Rate Adaptation configuration.

The key steps for configuring the Rate Adaptation feature are:

- To unambiguously determine the values for the set of client-specific parameters defined in this document.
- To translate these client parameters to the related parameters in the configuration.
- To operate a server with the derived configuration in an appropriate test-bed network.
- To use the test-bed network to refine the initial configuration according to the limitations of the network.
- To validate the configuration against a specified set of criteria.

The processes of testing and refinement are required because the underlying network may vary in any number of ways that are not necessarily obvious to the operator of the server. Understanding the interaction between the Rate Adaptation feature and a given network is essential to finalizing a Rate Adaptation configuration.

5.1.2 Client Specific Parameters

The first step in Rate Adaptation configuration for a new client is the determination of a set of client-specific parameters that have an impact on the performance of the Rate Adaptation feature. This section describes each of these parameters in detail, and defines the impact of each parameter on the operation of Rate Adaptation. The next section provides a mapping between the client parameters and the corresponding configuration parameters.

5.1.3 User Agent

The Rate Adaptation feature is configured discretely and activated for each unique client type. Client types are identified by their User-Agent string. The User-Agent is made available in the RTSP exchange used to establish the session. A given client implementation must send a valid and unique User-Agent string in order for Rate Adaptation to identify and associate a configuration with that client.

The User-Agent string can be determined by observing the RTSP exchange between the given client and the server.

For example:

```
DESCRIBE
rtsp://139.7.1.105:554/test/gprs2.mrc?msisdn=00491725551234&t=1086270520
672&p=credit&opcoid=OPCO&iver=3&gttime=1092737346&tokenname=X&guid=65fajk
10d26340268745000241000720&life=900&key=B7B3458538AF5E019B2CC5C9D4B14DFF
&usehostname RTSP/1.0

CSeq: 0

User-Agent: PVPlayer 3.4

Accept: application/sdp

x-wap-profile: http://wap.samsungmobile.com/uaprof/Z105UAProf.rdf

User-Network: UNKNOWN

DeviceInfo: MANUF=SAMSUNG;PROC=ARM;MEM=16MB;OS=PSOS;DISPLAY=

Timestamp: 1816238.750000
```

5.1.4 Client Technology

For the purpose of Rate Adaptation configuration the client technology may be divided into four categories:

1. 3GPP PSS Release 5 compliant clients
2. 3GPP PSS Release 6 compliant clients
3. RealNetworks 3GPP PSS Release 5 Annex G compliant mobile clients

The client technology of a given handset must be determined by contacting the handset vendor. The client technology impacts the general approach taken to Rate Adaptation configuration. However, RealNetworks 3GPP PSS Release 5 Annex G compliant clients are similar in their configuration requirements to 3GPP PSS Release 5 compliant clients.

5.1.5 RTCP RR Algorithm

The frequency of RTCP Receiver Reports may be determined by any one of three methods:

1. The SDP b=RR field.
2. The algorithm defined in IETF RFC 1889
3. An undefined variation of the algorithm defined in RFC 1889

The preferred method of determining which method is employed by a given client implementation is to contact the client technology vendor.

Rate Control state updates are driven by the RTCP Receiver Reports. Therefore, the frequency of the RTCP RRs determine the accuracy of the congestion control algorithm. Sparse RTCP RRs result in relatively long

durations during which the server does not have information from which to base adaptation decisions. This makes the server less quick to react to changes in network congestion.

The RTCP RR algorithm impacts the configuration of the round trip time (RTT) sample smoothing algorithm and the congestion control timeout sensitivity. If the client implements support for the SDP b=RR field, then the receiver report frequency must be defined.

5.1.6 Preroll Calculation

It has been found that not all clients observe the preroll value specified in the SDP for a given session. Some clients will institute the same preroll for all session, while others will employ an algorithm that is not readily ascertained simply by observing the client's behavior.

The client vendor must be contacted to determine the preroll methodology.

The preroll methodology is used by the buffer modeling component of the Rate Adaptation feature. The buffer model will not begin its playback timeline until after preroll has been completed. The server cannot estimate when preroll completes without knowledge of how preroll is calculated for the given session. If the server does not know the preroll algorithm for a given session, the client buffer model and server buffer model may not be synchronized. This makes the session prone to buffer overrun or under-run.

5.1.7 Buffer Model

During the course of operation, server-side Rate Control will modulate its rate in an effort to optimize the transmission rate for the network. The range of modulation is, in part, influenced by the size limitations of the client-side buffer. The 3GPP Release 5 PSS specification does not provide a facility for notifying the server of the state of the client buffer. Instead, the server must maintain a virtual model of the client-side buffer. The 3GPP Release 6 PSS specification provides some facility for notifying the server of the state of the client buffer through Next Application Data Unit (NADU) feedback.

Not all client post-network buffers are managed in the same fashion. Therefore, the server must be configured with parameters that indicate the characteristics of the client buffer model. The vendor of the client technology must be contacted to determine the client buffer model used. The following information must be requested from the client vendor:

- Does the client implement Annex G of the 3GPP Release 5 PSS?
- Does the client provide NADU feedback?
- If the client implements a non-standard buffer model, what is the size, in bytes, of the buffer chosen?
- Is the buffer size fixed, or does it vary?

5.1.8 Rebuffer Criteria

The downshift depth Rate Adaptation setting provides the operator with a way to configure Rate Adaptation to avoid rebuffer events by decreasing the encode rate of the transmitted session. In order for the downshift depth setting to have meaning, the client's method for determining a rebuffer event must be determined. This can only be determined by contacting the client vendor.

This method may be expressed as a simple threshold. For example, if the buffer level drops below some percentage of the preroll, or some percentage of the total buffer depth.

5.1.9 Network Support

A host handset for a given client may support just GPRS, or both UMTS and GPRS. The supported network type provides information on the upper bounds of the data rate that the client may support. The handset vendor provides information on the supported network types.

As of version 10.05 of the Helix Mobile Server, the operator will have to also be aware of certain aspects of the network configuration. The network configuration has an impact on the congestion control algorithm selected. Networks that employ large buffers along the link path may require that the Rate Control feature is configured to use an alternative form of congestion control that works to optimize the transmit rate while minimize the occupancy of network buffers. Large buffers in the link path are considered to be in excess of 1MB. Such buffer sizes are common in GPRS and UMTS networks that support a form of persistent retransmission at the link layer.

5.2 Creating a Baseline Configuration File

5.2.1 Overview

This section describes how to translate the information acquired for each of the required client parameters into a Rate Adaptation configuration file.

The Rate Adaptation configuration exists in two parts: 1) defining server configuration for each distinct client (User-Agent string) in the server configuration (rmserver.cfg) or in a separate User Agent Settings (.uas) file and 2) defining client specific values in the RDF file.

5.2.1.1 Server Configuration

The first part of a client configuration can be placed either in the Helix Mobile Server's configuration file (by default named "rmserver.cfg") or in a separate User Agent Settings file. In either case, the format of the file is XML. The Rate Adaptation feature is configured with an XML stanza of the following specification:

```
<List Name="MediaDelivery">
  <List Name="UserAgentProfiles">
    . . . [ Individual User Agent Settings ] . . .
  </List>
</List>
```

Alternatively this portion of the configuration may be stored in a separate file. These files are read from the User Agent Settings path, configurable via the following configuration entry in the rmserver.cfg file:

```
<List Name="MediaDelivery">
  <Var UASPath="examplepath"/>
  . . .
</List>
```

In cases where the settings are stored in a separate file, the top-level "MediaDelivery" and "UserAgentProfiles" lists are not required.

Regardless of where it is stored, a User Agent Settings entry is comprised of distinct sets of configuration parameters specific to a given client technology, keyed by a unique User-Agent string. One configuration may be shared between multiple User-Agents. The User-Agents supported by a configuration entry are defined in a "MatchUserAgents" list, as shown below.

Each of the User-Agent Profile list elements has the following general form:

```
<List Name="Example Configuration">
  <List Name="MatchUserAgents">
    <Var UserAgent_1="ExampleUserAgent"/>
  </List>
  <Var BaseProfile="Base Configuration"/>
  <Var UseServerSideRateAdaptation="always"/>
  <List Name="RateControl">
    <Var RTCPRRate="1.75%"/>
    <Var RTCPRRRate="3.75%"/>
    <Var UDPCongestionControlType="BCCC"/>
    <Var MaxSendRate="10000000"/>
    <Var FeedbackTimeout="15000"/>
  </List>
  <List Name="RateAdaptation">
    <Var DownshiftDepth="100%"/>
    <Var UpshiftDepth="200%"/>
  </List>
</List>
```

The example configuration will be referred to as 'ExampleConfiguration' in the remainder of this section. For the sake of brevity, XML configuration parameters will be written in hierarchical dotted format. For example, in the above configuration the ClientBufferPeriod variable would be written as:

```
ExampleConfiguration.ClientCapabilities.ClientBufferPeriod
```

5.2.1.2 RDF File

The second portion of a Rate Adaptation configuration is the RDF file. The relationship of the RDF file to a streaming session is defined in [3]. The client may provide a reference to this file through a URL in the RTSP exchange. If one is not specified then the settings may be retrieved from locally stored RDF files specified in the User Agent Settings, or from the User Agent Settings themselves. To specify a locally stored URI for the handset,

```
ExampleConfiguration.ClientCapabilities.CapabilityExchange.LocalRDF
```

must be set to the URI of the locally stored RDF file. Typically this URI references the local profile mountpoint, typically having the form <file://profiles/example.rdf>.

The User Agent Settings may also store the relevant information. The server retrieves two relevant values from the RDF file, the VideoDecodingByteRate and PreDecoderBufferSize, both defined in [3]. These values may be stored individually in:

```
ExampleConfiguration.ClientCapabilities.CapabilityExchange.VideoDecoding  
ByteRate  
ExampleConfiguration.ClientCapabilities.CapabilityExchange.VideoPreDecod  
erBufferSize
```

5.2.2 Procedure

5.2.2.1 User-Agent String

The User-Agent string is to be copied from the RTSP exchange to the configuration value:

```
ExampleConfiguration.MatchUserAgents.UserAgent_##
```

“##” represents a unique integer. Multiple user agents may share the same configuration by using sequentially increasing integers.

To always enable server-side Rate Adaptation for this client set the following variable to “always”:

```
ExampleConfiguration.UseServerSideRateAdaptation
```

To disable server-side Rate Adaptation to that client completely, set this variable to “never”. To allow the client to choose whether to use server-side Rate Adaptation (via “Helix-Adaptation” or “3GPP-Adaptation” headers defined in [3]) set this variable to “clientsselect”.



5.2.2.2 Client Technology

RealNetworks 3GPP clients that are compliant with Annex G may be configured like any other 3GPP compliant client.

Recall that the preroll time may not be indicated in the SDP for the session. Rather, it is specific to each client implementation. This value is a time in milliseconds. The suggested range is chosen as a balance between achieving the minimal amount of buffer required for a shift, which is the preroll, and a value that is not so long in duration such that potential Rate Adaptations are delayed.

RealNetworks mobile clients benefit from a Rate Control configuration that limits the rate the server may use when attempting an upshift according to the algorithm described earlier.

The following values can be set according to the guidelines provided in the section titled "Network Support":

```
ExampleConfiguration.RateControl.MaxSendRate
```

```
ExampleConfiguration.RateControl.MaxOversendRate
```

5.2.2.3 RTCP RR Algorithm

If the client implements support for SDP b=RR then the configuration variable:

```
ExampleConfiguration.RateControl.RTCPRRRate
```

can be set to indicate either the percentage of the initial media rate or the absolute maximum in bits per second for a given stream that will be used by the client for RTCP Receiver Report transmission. This value can be expressed as a percentage (i.e. "2.5%") or as a value in bits/s ("3000bps").

The bandwidth used according to the SDP b=RR specification does not change if the media rate is adapted.

This value must be chosen so that, given the initial media rate, the RTCP Receiver Reports are transmitted at an interval that approximates the expected RTT of the link.

If the SDP b=RR method is employed by a given client, then

```
ExampleConfiguration.RateControl.TFRC.RTTUseIIR
```

and/or

```
ExampleConfiguration.RateControl.BCC.RTTUseIIR
```

should be set to 0. This enables an aggressive smoothing algorithm that is only to be used when the RTT samples are very densely distributed in time.

If the SDP b= RR method is not employed by a given client, then

```
ExampleConfiguration.RateControl.TFRC.RTTUseIIR
```

and/or

```
ExampleConfiguration.RateControl.BCC.RTTUseIIR
```

should be set to 1, since the RTT samples will most likely be very sparse in time.

Also, since the RTCP receiver reports will arrive at an interval that is much larger than the RTT and since the Rate Control timeout mechanism is based on the anticipated time between RTCP receiver reports, it is important to set the timeout interval

```
ExampleConfiguration.RateControl.FeedbackTimeout
```

to a value that is less than 10x the expected RTT. This setting is the number of milliseconds that must elapse before a timeout is declared. RFC 1889 dictates that the minimum RTCP RR interval is 5000ms. Therefore, a value less than 50000ms is recommended. This ensures that Helix Mobile Server can take appropriate steps to ensure quality delivery in the event that RTCP receiver reports do not arrive frequently enough.

5.2.2.4 Preroll Calculation

If the client obeys the preroll time specified in the SDP, then no other action needs to be taken on the Rate Adaptation configuration. The server-side buffer model and the client-side buffer are working from the same set of assumptions about the preroll.

However, if the client does not obey the preroll time specified in the SDP, then the configuration element

```
ExampleConfiguration.ClientCapabilities.ClientBufferPeriod
```

must be set to the preroll time used by the client as expressed in milliseconds.



Preroll time is a concept that is only understood by RealNetworks clients. This value should only be configured for RealNetworks based clients.

5.2.2.5 Buffer Model

If a client does not support Annex G, then there are limitations as to what the server can reasonably do to manage the client buffer depth remotely. Future revisions of the 3GPP PSS will address this problem. In the meantime, client vendors should be strongly encouraged to implement the Annex G specification.

The most crucial consideration in buffer model configuration is the size of the buffer. If the size of the buffer, in bytes, is known, then that value can be entered in the corresponding configuration in:

```
ExampleConfiguration.ClientCapabilities.CapabilityExchange.VideoPreDecoderBufferSize
```

or in the UAS file as described in section 5.2.2.2.

If a fixed value is not known because the client uses a variable method for determining the buffer size, then there are three choices:

- Enter 1 as the value for `ExampleConfiguration.ClientCapabilities.CapabilityExchange.VideoPreDecoderBufferSize` in the User-Agent configuration. This allows the server to dynamically determine the buffer size according to the Annex G specification. The algorithm is specified in [3]. If results of this algorithm are consistent with the results of the algorithm used by the client to determine the buffer size for a given session, then 1 is a valid choice for this parameter.

- Given the client buffer size algorithm, select a fixed value that is sufficient for all data rates such that the value does not exceed the size for lowest data rate, and is sufficient to satisfy preroll for the highest data rate.
- Consider not using Rate Adaptation for the given client. If it is not possible for a valid client buffer size to be selected, then it is not possible for the server to prevent client buffer overflow. This will result in undefined behavior at the client.

A future revision of the server will allow for a dynamic buffer size algorithm.

5.2.2.6 Rebuffer Criterion

The upshift and downshift depths are expressed either as a percentage of the preroll time for the session or in absolute milliseconds. The downshift depth is the percentage of the preroll time below which a downshift event will be requested from the source file. The upshift depth is the percentage of the preroll time above which an upshift event will be requested from the source file. Although a rate shift may be requested, the request may not necessarily be satisfied. Shift requests are only satisfied when the transmission rate matches or exceeds an available rate in the source file.

An important distinction of the shift depth settings is that they are expressed as a function of the preroll, not the buffer size. For some clients, the preroll may equal the buffer size, while for others the buffer size may be greater than the preroll. Understanding how the client handles preroll and buffer size calculations is therefore essential to setting these values correctly.

It is also critical to understand the criterion for a rebuffer event at the client. The downshift depth should be set to at least 20 percent above the buffer level at which a rebuffer event would occur at the client. This threshold is maintained to allow for reaction time by the file source to a Rate Adaptation request. The reaction time is determined by the interval between key frames in the video stream. An encode rate change must always occur on a key frame boundary. If the key frame spacing happens to be such that client-side buffer drains faster than the shift can be executed, then the attempt to prevent a rebuffer event will have failed. Since the transmission rate is dynamic in time, it is not possible to predict with absolute certainty the extent of reaction time required to prevent a rebuffer event.

The following setting is expressed in milliseconds, and represents the time between when a shift request is made and when any subsequent requests may be honored.

```
ExampleConfigurationSession.RateAdaptation.StreamSwitchDelay
```

This setting defaults to 1000 milliseconds. The purpose of this setting is to allow the operator to define the maximal frequency of shift attempts. This value should be set to a multiple of the key frame interval of the expected content profile. This guideline is suggested because the key frame interval is the maximal frequency that encoded Rate Adaptations may be enacted. Setting `StreamSwitchDelay` to a value greater than 1x the key frame interval represents a tradeoff between reactivity and stability of the encoded rate.

5.2.2.7 Network Support

There are two additional Rate Adaptation parameters, dictated respectively by the network supported by the handset, and basic knowledge about the content selection supported by the handset.

The first is the maximum rate, expressed in bits per second, that the server will transmit to the given client. This should be set according to the network bearer supported by the given handset:

```
ExampleConfigurationRateControl.MaxSendRate
```



The second is the maximum rate allowed, expressed as a whole percentage of the current media rate being streamed:

```
ExampleConfiguration.RateControl.MaxOversendRate
```

The lesser of these two values is used is the maximal rate. These values should not be set so as to disallow shifting between data rates in a multi rate file. The OversendRate must always be able to reach the next higher rate, so that for each rate R_n except the highest one the following equation must be true:

$R_n * \text{MaxOversendRate}\% > R_{n+1}$ with R_{n+1} being the next higher rate

The following applies only to Helix Mobile Server version 10.05 or later:

If the network employs network buffers that exceed 1MB, which is common in GRPS and UMTS networks that support persistent retransmission at the link layer, then the congestion control algorithm should be set to BCC for clients that would otherwise be eligible for TFRC. Clients that use TFRC include 3GPP compliant clients that are not the legacy RealNetworks mobile client technology.

Setting the Rate Control type to the string value "BCC" enables BCC:

```
ExampleConfiguration.RateControl.UDPCongestionControlType
```

BCC is heavily parameterized, allowing the behavior of the BCC algorithm to be adjusted to a wide variety of network environments and/or scenarios. In general, however, the server operator should not modify the default values of the parameters for BCC. The configuration variables are listed in Appendix A and are for reference only.

5.3 Validating the Configuration

Validating the new handset configuration requires that a set of well-defined criteria be established. A fundamental success criterion can be broadly defined as transmission of the highest quality media given the constraints of the network capacity and variability. The bare minimum success criterion would be a transmission of media without rebuffering given changing network conditions.

Network variability is highly dynamic in mobile environments. Reducing this variability for the purpose of insuring that the configuration functions well in a baseline environment is therefore an appropriate initial step in validating a new configuration.

The validation and refinement cycle is driven by direct observation of streaming sessions on the target handset. Problems that are encountered through the course of configuration validation need to be investigated as a function of the given client, network, and server technologies. Troubleshooting the client and network components is out of the scope of this document.

The server provides a Rate Adaptation tracing feature that can be used to track the operation of a Rate Adaptation session through the course of action. Using Rate Adaptation traces is essential to understanding the impact of the configuration settings chosen in the previous sections on the behavior of the system at run-time.

5.3.1 Run Time Tracing

To enable tracing set the following configuration variable to "1":

```
ExampleConfiguration.DebugOutput
```



In cases where TFRC is used, this will cause the server to write a file with the naming convention:

```
tfrc_<rtsp_session_id>_<stream_number>.txt
```

to the root directory of the Helix Mobile Server installation. <rtsp_session_id> corresponds to the RTSP session identifier string associated with the given file. Similarly, the <stream_number> variable corresponds to the RTSP stream number within the session that associated with the given file.

Each Rate Control log contains variable and fixed information used by the TFRC algorithm during the course of operation. All TFRC log files are pre-pended with a header that represents the configuration information that is relevant to this session.

For example:

```
RTT IIR Filter: Not Used

Rate Increase Limit Scalar: 1.2500
```

The remainder of the file consists of a sequence of blocks of statistics. Each entry of each block has the same timestamp. The timestamp represents the time at which the statistics became available. In the case of RTP sessions, the timestamps represent the time at which an RTCP Receiver Report arrived. The time units are milliseconds relative to the beginning of the session.

For example:

```
1030 65154 media_rate
1030 65154 tfrc_rate
1030 66192 actual_rate
1030 0.012 recvd_rate
1030 1 state
1030 0 num_lost
1030 1 num_recv
1030 0.00052 loss
1030 428.00903 rtt
1030 428.009 raw
1030 20.688 sq_mean
```

It is therefore possible to parse for specific statistics and to graph the results.

For example the Bash shell command:

```
grep "tfrc_rate" tfrc_1234_0.txt > tfrc
```



would result in a file named "tfrc" that contains just the time series variation of the transmission rate. This file can be used by applications such as gnuplot or Excel to readily graph time series data for comparison and analysis.

Each of the statistic entries are defined as follows:

- `media_rate`: The current encode rate of the stream.
- `tfrc_rate`: The rate that the congestion control algorithm is transmitting data.
- `actual_rate`: `actual_rate`: The current measured outbound send rate.
- `recvd_rate`: The rate at which the client receives data (according to server calculation, see section 3.2 "Rate Control").
- `num_lost`: The number of packets lost in the last reporting interval.
- `num_recv`: The number of packets received in the last reporting interval.
- `loss`: The TFRC loss event ratio.
- `rtt`: The statistically smoothed RTT.
- `raw`: The RTT-derived sample from the current RTCP Receiver Report.

In Helix Mobile Server version 10.05 or later the BCC congestion control algorithm is available.

Enabling tracing as described earlier in this section will cause the server to write a file with the naming convention for BCC tracing:

```
bcc_<rtsp_session_id>_<stream_number>.txt
```

to the root directory of the Helix Mobile Server installation. `<rtsp_session_id>` corresponds to the RTSP session identifier string associated with the given file. Similarly, the `<stream_number>` variable corresponds to the RTSP stream number within the session that associated with the given file.

Each log contains variable and fixed information used by the BCC algorithm during the course of operation. All BCC log files are pre-pended with a header that represents the configuration information that is relevant to this session.

For example:

```
RTT Filter Algorithm:
Increase Coefficient:
Decrease Coefficient:
Increase Exponent:
Decrease Exponent:
Increase Threshold:
Decrease Threshold:
Lower Limit:
Target Ratio:
```

The remainder of the file consists of a sequence of blocks of statistics. Each entry of each block has the same timestamp. The timestamp represents the time at which the statistics became available. In the case of RTP sessions, the timestamps represent the time at which an RTCP Receiver Report arrived. The time units are milliseconds relative to the beginning of the session.

For example:

```
<time_stamp_ms>   send_rate
<time_stamp_ms>   media_rate
<time_stamp_ms>   actual_rate
<time_stamp_ms>   rtt_filter
<time_stamp_ms>   rtt_raw
<time_stamp_ms>   num_recvd
<time_stamp_ms>   num_lost
<time_stamp_ms>   inc_coef
```

The meaning of each of the fields is as follow:

- `send_rate`: The rate in bits per second at that the congestion control suggests the MDP transmit.
- `media_rate`: The rate in bits per second of the media behind transmitted in the present stream.
- `actual_rate`: `actual_rate`: The current measured outbound send rate.
- `rtt_filter`: The smoothed RTT in milliseconds.
- `rtt_raw`: The unfiltered RTT sample from the current client report in milliseconds.
- `num_recvd`: The number of packets received as reported in the current client report. This value is not cumulative for the duration of the session.
- `num_lost`: The number of packets lost as reported in the current client report. This value is not cumulative for the duration of the session.
- `inc_coef`: The current increase coefficient from the rate increase function. This value is dynamically calculated on each congestion control interval to insure that the user configured target decrease:increase rate is met. The decrease:increase ratio defaults to 4, and is the ratio of the magnitude of rate decreases to the magnitude of rate increases for a given set of BCC parameters and the current sending rate.
- `<time_stamp_ms>`: The number of milliseconds since the start of the session.

It is therefore possible to parse for specific statistics and to graph the results.

For example the Bash shell command:

```
grep "send_rate" bcc_1234_0.txt > bcc
```

would result in a file named "bcc" that contains just the time series variation of the transmission rate. This file can be used by applications such as gnuplot or Excel to readily graph time series data for comparison and analysis.

5.3.2 Run Time Tracing for Buffer Control and Stream Switching

To enable tracing for the Rate Adaptation set the following configuration variable to "1":

```
ExampleConfiguration.DebugOutput
```

This will cause the server to write a file with the naming convention:

```
ratemgr_<rtsp_session_id>.txt
```



to the root directory of the Helix Mobile Server installation. <rtsp_session_id> corresponds to the RTSP session identifier string associated with the given file.

Each Rate Adaptation log contains variable and fixed information used by the buffer management algorithm during the course of operation. All Rate Adaptation log files are prepended with a header that represents the configuration information that is relevant to this session:

```
PreDecodeBufferSize: 30000
PreDecodeBufferPeriod: 7000
BufferLowWaterMark: 3500
BufferHiWaterMark: 10500
```

The definition of these fields is as follows:

- `PreDecodeBufferSize`: The size, in bytes, of the client buffer model as derived from the `VideoPreDecodeBufferSize` in the User Agent Settings or Client Profile (RDF).
- `PreDecodeBufferPeriod`: The time, in milliseconds, that represents the preroll for this client and content as encoded in the content file or configured in the User Agent Settings. Rate Adaptation will not evict packets from the buffer model until the buffer is filled to the depth of this value.
- `BufferLowWaterMark`: The downshift depth, in milliseconds of buffered data. Calculated from the `PreDecodeBufferPeriod` times the percentage as configured in the User Agent Settings.
- `BufferHiWaterMark`: The upshift depth, in milliseconds of buffered data. Calculated from the `PreDecodeBufferPeriod` times the percentage as configured in the User Agent Settings.

The remainder of the file consists of a time sequence of statistics. Each entry has a timestamp. The time units are milliseconds relative to the beginning of the session. The time stamp for each entry is followed by state information, the current rendering time, and the stream number.

It is possible to parse for specific statistics and to graph the results.

For example the Bash shell command:

```
grep "0 buffdepth" ratemgr_1234_0.txt > buffer_depth_0
```

will result in a file named "buffer_depth_0" that contains time series variation of the buffer depth for stream 0 across multiple events. This file can be used by applications such as gnuplot or Excel to readily graph time series data for comparison and analysis.

Each packet processed is logged in a line with roughly the same format:

```
playback_time state queue_time streamid action timestamp of packet buffer fill level
```

where:

<i>playback_time</i>	The time in milliseconds since the start of the streaming session.
<i>state</i>	The current state of the buffer model: init, predecode or decode.
<i>queue_time</i>	The current render time in the queue. All packets earlier than this time are evicted.
<i>streamid</i>	The stream id the packet belongs to (e.g 0=video, 1=audio)



action What has been done with the packet (see below)
timestamp of packet The playback time of the stream the packet contains data for
buffer fill level The fill level of the buffer model after the operation, in bytes

For example:

```
InsertIntoQueue: State=init->predecode, m_ulPTS1=33
. . .
938 predecode 33 0 Queued_packet seq=12 ts=834 size=642 -> bufsize=8568
buffdepth=801
. . .
InsertIntoQueue: State: predecode->decode, m_ulPTS1=33, time 1101
. . .
1360 decode 190 0 Evicted_packet seq=2 ts=167 size=19 -> bufsize=11152
. . .
15203 decode 10528 0 Stream_blocked seq=25 ts=12563 size=563
bufsize=40397 retry=300
. . .
15203 decode 0 Stream_blocked 10563 40397
. . .
15707 decode 10892 0 Expired_packet seq=32 ts=9062 size=367 ->
bufsize=10575
. . .
16345 decode 11729 0 NADU_Feedback playout_delay=4526 NSN=32 NUN=34
FBS=72 (4608) HSNR=40
16727 UpShift
. . .
20344 DownShift
```

Each of the entries are defined as follows:

- *State=init->predecode*: Logs that the buffer model entered the predecode state, that is, queueing packets but not evicting them. The initial timestamp of the first packet is 33ms.
- *Queued_packet*: Indicates when a packet for a given stream has entered the buffer model. In the example above, at session time 938 ms the packet itself has a timestamp of 834 ms. The buffer depth

is 801 milliseconds; the current packet's timestamp minus the first packet's timestamp. The buffer contains 8568 bytes of data. In the pervious report (not shown), the buffer size was 7926 bytes.

- `State: predecode->decode`: Logs that the buffer model entered decode state, that is, packets start now to be evicted. The next packet queued will be the packet with timestamp 1101 ms.
- `Evicted_packet`: Indicates when a packet from a given stream has left the buffer model to be "decoded". In the example above, the current queue time (190) is greater than the packet's timestamp (167). The buffer contains 11152 bytes of data, reflecting the reduction in size from evicting this packet. In the previous report (not shown), the buffer size was 11171 bytes.
- `Expired_packet`: Indicates that Helix Mobile Server perceives that a packet from a given stream should have left the buffer model to be "decoded", but a NADU update indicating that the packet has been "decoded" has not been received. In the example above, the current queue time (10892) is greater than the packet's timestamp (9062). The buffer contains 10575 bytes of data, reflecting the reduction in size from evicting this packet. In the previous report (not shown), the buffer size was 10942 bytes.
- `Stream_blocked`: Indicates that transmission is blocked because the buffer model is full. Logs that Rate Adaptation wanted to queue a packet, but could not because there was no room left for the packet in the buffer model. In the example above, this happened at playback time 15203, with the packet's timestamp being 12563. The buffer contains 40397 bytes, while the buffer size (as taken from the header) is 40960 bytes. The packet in question is obviously larger than $(40960-40397=)$ 563 bytes, so it does not fit in the free room left of the buffer. When Rate Adaptation shows `Stream_blocked` messages, the server effectively limits its delivery rate to the actual media rate.
- `NADU_Feedback`: Indicates the buffer model has received NADU feedback. More information on NADU is available in [3]. In this example ,the NADU feedback indicates that the playout delay is 4526ms, the sequence number of the next packet to be decoded is 32, the amount of free space in the buffer is 4608 bytes (which NADU reports as 72 x 64 byte blocks). The next unit number is reported as unit 34, and the highest sequence number on a packet received so far is 40.
- `UpShift`: Indicates that an upshift request has been made. In the example above, at playback time 16727, The Rate Adaptation module requested an upshift. No indication is given whether the upshift actually has been performed.
- `DownShift`: Indicates that a downshift request has been made. In the example above, at playback time 20344, The Rate Adaptation module requested a downshift. No indication is given whether the downshift actually has been performed.

6 CONTENT ENCODING GUIDELINES

6.1 Recommendations

The way the content is encoded highly contributes to the overall user experience of the stream. Tests showed that the included media rates should not be too far from each other in order to facilitate upshifting.

The following media rates combined in one presentation ensure no media rate is more than 140% of the next lower media rate, giving a reasonable density of media rates in one file.

For GPRS rates, use content with the following media rates:

- 15, 20, 25, and 30 kbps

For UMTS rates, use content with the following media rates:

- 75, 85, 95, 105, 115 kbps

If the content is intended to be used in both GPRS and UMTS environments, use:

- 15, 20, 25, 30, 55, 75, 85, 95, 105, 115 kbps

6.2 Considerations

As with all other aspects of Rate Control configuration, correct content encoding parameters are a function of a number of variables. These variables are primarily couched in the interaction between the Rate Control feature and the given network. Since radio networks can vary greatly in their characteristics, developing stable definitions of network parameters does require experimentation on target networks to determine the stable operating points of the media transmission system on the given networks.

As Rate Adaptation operates, the transmission rate will oscillate. These oscillations are the result of congestion control and client buffer management working together over the dynamics of the underlying network. The frequency of RTCP receiver reports and the TFRC or BCC algorithms determine the frequency of the rate oscillation. TFRC and BCC both will attempt to damp oscillations over time.

Short-term variations in rate translate to stable operating rates over the longer term. The stable operating rates reached by the system will depend entirely on the network. Therefore, the operator must determine, using single rate content, what the stable transmission rate is for given network bearers.

This process should be conducted with the trace logging turned on for the TFRC or BCC Congestion Control modules. The server must also be configured for the client against which the test will be performed. The configuration should be done according to the process defined in this document.

Single rate content encoded at a rate that is known to be close to the maximal rate of the network should be used. The content should be at least 5 minutes in length to provide enough time to collect sufficient data samples at the server. After a test run the TFRC or BCC log file for the test run is collected. The log file is then parsed to determine the average transmission time determined by TFRC or BCC and the received rate as calculated from the RTCP receiver reports. The averaging window should be equal to the preroll of time of the session.

Once this average has been determined with sufficient statistical accuracy, a stable operating point for the transmission system over the given network can be noted. Rates in a multirate file can then be mapped to these operating rates for the target bearers, and the Rate Adaptation configuration file can then be tuned to allow for migration between rates by modifying the configuration value:

```
ExampleConfiguration.RateControl.MaxOversendRate
```

There is a quality overhead associated with media adaptations. At a shift point, the user may experience transient frozen or distorted video for a brief period of time that is determined by the encoding parameters for the given content. It follows that rate adaptive content should be tuned to minimize the frequency of media shifts while allowing for optimal responsiveness to the expected type of varying network conditions.

Having data rates in a multirate file that are between stable operating points comes at the cost of increased rate switches, and is not suggested. Instead the operator should rely on the fact that the server will attempt to use the client side buffer to send the media data ahead of real time, allowing for increased resiliency during intermittent congestion events. Rate switching in response to transient events will likely be too costly on the user experience. Rate switching should, if possible, be reserved for network hand over events or time slot reallocation.

A potential use for intermediary data rates between stable operating points is to constrain the rate over which Rate Control may oscillate. If, for example, the operator does not want to use a MaxOversendRate that is very

high, therefore allowing a lot of variability in the sending rate, the operator can reduce the required scalar by reducing the difference in between data rates in a multirate file.

7 PERFORMING A TEST

7.1 What to collect from a performed test

- rmserver.cfg
- all user agent settings (.uas) files
- all client profiles (*.rdf files)
- the Rate Adaptation trace file ratemgr*.txt
- the TFRC or BCC trace file
- rmaxcess.log
- rmerror.log
- the server's standard output
- dauc.log
- rtspevents.log
- rtsp_messages.log (RTSP debug log file)
- A user experience QoS log, including rebuffers, freezes, artifacts, player abortions and other noticeable events seen watching the stream
- Snoops of the traffic (taken with snoop/ethereal/tcpdump, from as many points as possible)

8 APPENDIX A – VARIABLE DEFINITIONS

The following table defines a subset of the available configuration variables for the server-side Rate Adaptation feature.

Configuration entry	Type	Units	Legal values	Description
X.*.MatchUserAgents.Use rAgent_##	Str	n/a	n/a	Expressions against which to match given User-Agent strings from RTSP Headers. There may be more than one expression per profile, allowing multiple User-Agent types to use the same profile. Each expression exists within the "MatchUserAgents" list and has the format "UserAgent_01", "UserAgent_02", etc.
X.*.UseServerSideRateAd aptation	Str	n/a	always, never, clientse lect	Indicates whether server side Rate Adaptation is used for this User-Agent. If set to "clientselect", server side Rate Adaptation will be used only if the client reports that it supports Helix Rate Adaptation or 3GPP Rate Adaptation. If set to "always", the server will use ERA regardless of what the client claims to support. If set to "never", server side Rate Adaptation will be disabled for this client.
X.*.BaseProfile	Str	n/a	n/a	The name of a base profile which this profile inherits settings from. This setting allows User-Agents to share common characteristics by inheriting them from a base profile, then add settings specific to the particular User-Agent without duplicating entire profiles.
X.*.DebugOutput	Int	Bool	0, 1	Enables debug tracing, including but not limited to dumping of Rate Adaptation, TFRC and BCC logs.
X.*.RateAdaptation.Defa ultMediaRate	Int	Bps	> 0	The default initial media rate for this session to select from a group of media rates in a multirate file if the client provides no indication of available bandwidth.
X.*.RateAdaptation.Down shiftDepth	Int	% (of preroll) or ms	0 – N	Downshift depth for a User-Agent. The server uses a model of the client's buffer to determine when to shift. A shift upward between rates occurs when the client buffer reaches the upshift depth, and a downshift occurs when the client buffer reaches the downshift depth. These settings determine the extent to which the server is conservative or liberal in its attempt to adapt the media up or down. A "%" value sets the depth as a percentage of the preroll. If this value is set to "80%" then, with a 1000ms preroll, a downshift will occur if the client buffer depth falls below 800ms. This value can also be explicitly set to a ms value.
X.*.RateAdaptation.Upsh iftDepth	Int	% (of preroll) or ms	0 – N	Upshift depth for the client, if the client's buffer depth goes above this depth, this signals to the server that an upshift may be possible. A "%" value sets the depth as a percentage of the preroll. If this value is set to "150%" then, with a 1000ms preroll, a downshift will occur if the client buffer depth falls below 1500ms.



Configuration entry	Type	Units	Legal values	Description
X.*.RateAdaptation.EnableStepwiseUpshift	Int	Bool	0, 1	Stepwise upshifting will cause the Rate Adaptation system to cycle through each media rate between the current media rate, and the maximum maintainable rate, allow for a more conservative (slower) up-shifting method. If stepwise upshifting is disabled and the Rate Adaptation system deems an upshift is possible, it will shift immediately to the highest sustainable rate.
X.*.RateAdaptation.EnableStepwiseDownshift	Int	Bool	0, 1	Stepwise downshifting will cause the Rate Adaptation system to cycle through each media rate between the current media rate, and the minimum maintainable rate, allow for a more aggressive (slower) downshifting method. Enabling stepwise downshifting allows the Rate Adaptation system to maintain a higher encoding rate through transient congestion, but makes the Rate Adaptation system slower to adapt the rate in cases where bandwidth drops sharply for a longer term.
X.*.RateAdaptation.StreamSwitchDelay	Int	ms	0 - N	Minimum time in milliseconds after a rate shift is made (either down or up) before another rate shift can be made (either down or up).
X.*.RateAdaptation.ClientAdaptation.Enable3GPPAdaptation	Int	Bool	0, 1	Enables and disables 3GPP-Adaptation. Enabling instructs the client to send, and the server to use, Helix-Adaptation feedback headers. Disabling instructs the client to not send Helix-Adaptation feedback, and causes the server to ignore any Helix-Adaptation feedback it receives.
X.*.RateAdaptation.ClientAdaptation.EnableHelixAdaptation	Int	Bool	0, 1	Enables and disables Helix Rate Adaptation. Enabling instructs the client to send, and the server to use, Helix-Adaptation feedback headers. Disabling instructs the client to not send Helix-Adaptation feedback, and causes the server to ignore any Helix-Adaptation feedback it receives.
X.*.RateAdaptation.ClientAdaptation.TargetTimeDownshiftDepth	Int	% (of target time) or ms	> 0	Downshift depth for a User-Agent. Similar to "X.*.RateAdaptation.DownshiftDepth", above, but based on percentage of target time instead of preroll. Target time is provided by 3GPP-Adaptation headers[3] for RTP or Helix-Adaptation headers for RDT.
X.*.RateAdaptation.ClientAdaptation.TargetTimeUpshiftDepth	Int	% (of target time) or ms	> 0	Upshift depth for a User-Agent. Similar to "X.*.RateAdaptation.UpshiftDepth", above, but based on percentage of target time instead of preroll. Target time is provided by 3GPP-Adaptation headers[3] for RTP or Helix-Adaptation headers for RDT.
X.*.RateAdaptation.ClientAdaptation.EnableClientReportedUpshift	Int	Bool	0, 1	Enable upshifting when 3GPP-Link-Char or Bandwidth header information says there is sufficient bandwidth to maintain higher rate stream.
X.*.RateAdaptation.ClientAdaptation.EnableClientReportedDownshift	Int	Bool	0, 1	Enable downshifting when 3GPP-Link-Char or Bandwidth header information says there is



Configuration entry	Type	Units	Legal values	Description
ntReportedDownshift				insufficient bandwidth to maintain current stream.
X.*.RateControl.RtcpRRRate	Int	% or bps	> 0	The amount of bandwidth to use for RTCP Receiver reports. Helpful to trade overhead costs for granularity vs. responsiveness. This value can be expressed as Bps or a percentage of session bandwidth.
X.*.RateControl.RtcpRSRate	Int	% or bps	> 0	The amount of bandwidth to use for RTCP Sender reports.
X.*.RateControl.UDPCongestionControlType	Str	n/a	TFRC or BCC	Type of Congestion Control scheme to be used for the client.
X.*.RateControl.MaxOversendRate	Int	%	> 100	Mechanism to set the max allowable rate for Rate Control as a percentage of the media rate e.g. 125% - This is the percentage of the session media rate that the server may maximally transmit. This value determines the ability of the server to up shift between rates, as it places an upper bounds on the transmission rate based on the currently transmitted rate.
X.*.RateControl.MaxSendRate	Int	bps	> 0	This is the maximum rate that Rate Control may send at in the course of a streaming session. It is an estimate of the rate allowable by the channel. This can be set arbitrarily high, but the cost is that Rate Control may periodically over shoot the real channel rate causing loss.
X.*.RateControl.InitialOversendRate	Int	%	>100	The start rate used by Rate Control as a percentage of the media rate. This is the maximum percentage over the session media rate that the server may initially begin transmitting. This value defaults to 400% in order to mimic TurboPlay behavior when the MDP is used.
X.*.RateControl.FeedbackTimeout	Int	ms	>0	Absolute time in milliseconds without having received feedback before the server abandons Rate Control and delivers at the media rate instead.
X.*.RateControl.DeliverBelowLowestMediaRate	Int	Bool	0, 1	Determines whether the server will deliver media below the lowest media rate in the stream if Rate Control deems this to be appropriate.
X.*.RateControl.BCC.RTTUseIIR	Int	Bool	0,1	Indicate whether to use a responsive infinite impulse response (IIR) filter for RTT filtering. TRUE trades smoothness for responsiveness.
X.*.RateControl.BCC.TimeoutTransmission	Int	Bool	0,1	Determines whether or not Congestion Control will respond to timeout events at all. Enabled means that the server will stop sending packets when a feedback timeout occurs; Disabled means that the server will cut its sending rate in half for each timeout event.
X.*.RateControl.BCC.DecreaseCoefficient	Int	Integer	2-32 Default: 8	The constant coefficient used in the BCC rate decrease function. See [14] for further details about the BCC rate decrease function.



RATE CONTROL FOR MOBILE NETWORKS

X.*. RateControl.BCC.IncreaseCoefficient	Int	Integer	16000-512000	The constant coefficient used in the BCC rate increase function. See [14] for further details about the BCC rate increase function. Default: 512
X.*.RateControl.BCC.DecreaseExponent	Float	Exponent	0-1	The exponential term used in the BCC decrease function. See [14] for further details about the BCC rate decrease function. The decrease exponent plus increase exponent must not be greater than 1. Default: 0.5
X.*.RateControl.BCC.IncreaseExponent	Float	Exponent	0-1	The exponential term used in the BCC increase function. See [14] for further details about the BCC rate increase function. The decrease exponent plus increase exponent must not be greater than 1. Default 0.5
X.*.RateControl.BCC.DecreaseThreshold	Int	ms	1-N	The number of milliseconds the RTT must increase before the decrease function is applied. Default: 5
X.*.RateControl.BCC.IncreaseThreshold	Int	ms	0-N	The number of milliseconds the RTT must decrease before the increase function is applied. Default: 0
X.*.RateControl.BCC.LowerLimit	Int	bps	128-N	The lowest rate at which the binomial congestion control algorithm will send data. Default: 2048
X.*.RateControl.BCC.TargetRatio	Int	Integer	2-N	The desired ratio of decrease magnitude to increase magnitude that is maintained by dynamically adjusting the increase coefficient. Default: 4
X.*.ClientCapabilities.ClientBufferPeriod	Int	ms	0-N	Value used by the client buffer model to estimate the when the client begins playback, and starts using data from its buffer. If the client does not obey the preroll time specified in the SDP, this value must be set to the preroll time used by the client as expressed in milliseconds
X.*.ClientCapabilities.BandwidthMultiplier	Int	%	>100	Adjusts the reported bandwidth in b=AS header of SDP file.
X.*.ClientCapabilities.BandwidthProtocol	Str	n/a	tcp, udp	Used to determine protocol bandwidth overhead reported in SDP.
X.*.ClientCapabilities.	Int	Integer	0 - N	Ratio of data packets to RDT buffer info reports. A



RDTBufferInfoRatio				lower number means more buffer info reports (less packets per report).
X.*.BufferUsageLimit	Int	%	0 - 100	Sets the upper limit on the percentage of the Client's buffer the Server will attempt to fill. The actual usage state of the client buffer is not known exactly by the Server. This provides a margin of safety to prevent the Server from overflowing the Client buffer.
X.*.ClientCapabilities.CapabilityExchange.LocalRDF	Str	n/a	URI	The path and filename of a locally stored RDF file, should one exist for the associated handset.
X.*.ClientCapabilities.CapabilityExchange.RetrieveXWAPProfiles	Int	Bool	0, 1	Enables or disables the retrieval of the X-WAP profile from the URL included in the x-wap-profile header/s sent by the client. If disabled, the local values alone will be used.
X.*.ClientCapabilities.CapabilityExchange.VideoPreDecoderBufferSize	Int	Bytes	0-N	This attribute signals if the optional video buffering requirements defined in Annex-G are supported. It also defines the size of the hypothetical pre-decoder buffer defined in Annex-G. For more information please reference section 5.2.3.2.1 of [3].
X.*.ClientCapabilities.CapabilityExchange.VideoDecodeByteRate	Int	bps	>=8000	If Annex G is supported, this attribute defines the peak decoding byte rate the PSS client is able to support. The values are given in bytes per second and shall be greater than or equal to 8000. According to Annex G, 8000 is the default peak decoding byte rate for the mandatory video codec profile and level (H.263 Profile 0 Level 10). For more information please reference section 5.2.3.2.1 of [3].

9 APPENDIX B - SAMPLE RDF FILE

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns"
  xmlns:ccpp="http://www.w3.org/2000/07/04-ccpp"
  xmlns:prf="http://www.wapforum.org/profiles/UAPROF/ccppschemata-20010330"
  xmlns:pss5="http://www.3gpp.org/profiles/PSS/ccppschemata-PSS5">
  <rdf:Description rdf:about="http://www.bar.com/Phones/Phone007">
    <ccpp:component>
      <rdf:Description ID="HardwarePlatform">
        <rdf:type
rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschemata-20010330#HardwarePlatform"/>
        <prf:Model>Z105</prf:Model>
        <prf:Vendor>Samsung</prf:Vendor>
      </rdf:Description>
    </ccpp:component>
    <ccpp:component>
      <rdf:Description ID="Streaming">
```

```
<rdf:type rdf:resource=" http://www.3gpp.org/profiles/PSS/ccppschem-
PSS5#Streaming"/>
    <pss5:VideoPreDecoderBufferSize>30000</pss5:VideoPreDecoderBufferSize>
        <pss5:PssVersion>3GPP-R5</pss5:PssVersion>
    </rdf:Description>
</ccpp:component>
</rdf:Description>
</rdf:RDF>
```

10 APPENDIX C – SAMPLE MDP CONFIG

```
<List Name="PacketVideoClients">
  <Var BaseProfile="Default"/>
  <Var UseServerSideRateAdaptation="always"/>
  <List Name="RateControl">
    <Var UDPCongestionControlType="TFRC"/>
    <List Name="TCP">
      <Var HonorMaxSendRate="1"/>
    </List>
  </List>
  <List Name="RateAdaptation">
    <Var DefaultMediaRate="32000"/>
    <Var DownshiftDepth="50%"/>
  </List>
  <List Name="ClientCapabilities">
    <Var ClientBufferPeriod="5000"/>
    <List Name="CapabilityExchange">
      <Var VideoPreDecoderBufferSize="1"/>
      <Var VideoDecodeByteRate="0"/>
      <Var PSSVersion="3GPP-R5"/>
    </List>
  </List>
</List>

<List Name="Z105">
  <Var BaseProfile="PacketVideoClients"/>
  <List Name="MatchUserAgents">
    <Var UserAgent_1="PVPlayer 3.4"/>
  </List>
  <List Name="RateControl">
    <Var MaxSendRate="132000"/>
    <Var MaxOversendRate="200%"/>
  </List>
  <List Name="RateAdaptation">
    <Var DownshiftDepth="20%"/>
    <Var UpshiftDepth="120%"/>
  </List>
  <List Name="ClientCapabilities">
    <Var ClientBufferPeriod="7000"/>
    <List Name="CapabilityExchange">
      <Var VideoPreDecoderBufferSize="3000"/>
      <Var VideoDecodeByteRate="0"/>
      <Var PSSVersion="3GPP-R5"/>
    </List>
  </List>
</List>
```

</List>
</List>
</List>

11 APPENDIX D – DATA REQUIRED FROM CLIENT MANUFACTURER

This section outlines the data that must be collected from the client manufacturer in order to enable the terminal to use the server-side Rate Control feature.

1. Hardware Platform Details

Vendor Model information. Egs., Nokia 6600, Sony Ericsson P800

2.

3. Client Software Details

Name: eg., RealOnePlayer

Version: eg., 3.2.23

4. User-Agent Details

Identify the User-Agent string sent by the client to the server (can be observed through network traces).

Eg., Nokia 3650 User Agent string="RealOnePlayer/3.2.23_epoc_series60_thumb"

5. Codecs Supported

List of all video and audio codecs supported.

6. Mode of Phone

GPRS, UMTS, CDMA, WCDMA, or dual mode?

7. Streaming Attributes (Specified in TS 26.234)

VideoPreDecoderBufferSize _____

VideoDecodingByteRate _____

8. 3GPP Compliance

Release 5 _____ Yes _____ No

Annex G _____ Yes _____ No

Release 6 _____ Yes _____ No

9. Buffering Period

Identify the typical buffering period before the client starts playback.

10. Receiver Report Frequency

Identify whether the client supports b=rr sent in the SDP _____ Yes _____ No

If not what is the frequency of the Receiver reports (Units: number of RRs sent per second)?

12 REFERENCES

[1] 3GPP TS 22.233: "Transparent End-to-End Packet-switched Streaming Service; Service aspects; Stage 1 "

[2] 3GPP TS 26.233: " Transparent end-to-end packet switched streaming service (PSS); General description".

[3] 3GPP TS 26.234: "End-to-end transparent streaming Service; Protocols and codecs".

[4] IETF RFC 2326: "Real Time Streaming Protocol (RTSP)", Schulzrinne H., Rao A. and Lanphier R., April 1998.

- [5] IETF RFC 1945: "Hypertext Transfer Protocol - HTTP/1.0", Fielding R. et al., May 1996.
- [6] IETF RFC 2327: "SDP: Session Description Protocol", Handley M. and Jacobson V., April 1998.
- [7] IETF RFC 3267: " RTP payload format and file storage format for the Adaptive Multi-Rate (AMR) Adaptive Multi-Rate Wideband (AMR-WB) audio codecs ", March 2002.
- [8] IETF RFC 3016: "RTP Payload Format for MPEG-4 Audio/Visual Streams", Kikuchi Y. et al., November 2000.
- [9] IETF RFC 2429: "RTP Payload Format for the 1998 Version of ITU-T Rec. H.263 Video (H.263+)", Bormann C. et al., October 1998.
- [10] IETF RFC 1889: "RTP: A Transport Protocol for Real-Time Applications", Schulzrinne H. et al., January 1996.
- [11] IETF RFC 3556: "Session Description Protocol (SDP) Bandwidth Modifiers for RTP Control Protocol (RTCP) Bandwidth", Casner S., July 2003.
- [12] 3GPP TS 26.244: " Transparent end-to-end packet switched streaming service (PSS); 3GPP File Format".
- [13] IETF RFC 3448: "TCP Friendly Rate Control (TFRC): Protocol Specification"
- [14] Bansal, et al., "Binomial Congestion Control Algorithms", M.I.T. Laboratory for Computer Science, 2001

13 DOCUMENT CONTROL

13.1 Contributions

The following RealNetworks personnel have collaborated on this document:

Contributor	Division/Group
Damon Lanphear	Helix Products-BPS Business Products Development
Uma Boddeti	Helix Products-BPS Business Products Development
Darrick Lew	Helix Products-BPS Business Products Development
Jason Simpson	Helix Products-BPS Business Products Development Operations
Merritt Bettineski	Helix Products-BPS Business Products Development Operations
Peter Kanzow	Helix Products-Germany
Jim Denenny	Carrier-England
Mike McCrory	Carrier-England

13.2 Version Control

Version number	Date of version	Entered by	Summary
1.0	August 11, 2004	Katie Leggett	Initial draft
1.1	August 18, 2004	Katie Leggett	Revised draft with contributions on creating new configurations from Uma
1.2	August 23, 2004	Katie Leggett	Including contributions from Merritt Bettineski, Damon Lanphear and Peter Kanzow
1.3	August 30, 2004	Katie Leggett	Finalized draft for review by team
1.4	September 1, 2004	Katie Leggett	Final draft
1.5	October 15, 2004	Jordan Friedman	Fixes of grammar errors. Inclusion of corrected text drafter by Peter Kanzow on Timeout handling configuration parameters and the NetworkAffinity parameter used for legacy microcore clients.
1.6	January 7, 2005	Damon Lanphear	Add documentation of Binomial Congestion Control introduced in Helix Mobile Server version 10.05
1.7		Darrick Lew	Update document to reflect changes made for Helix Mobile Server version 12.0.
1.8	February 4, 2008	Scott MacKenzie	Branding changes

