Broadcast and Multicast Communication Enablers for the Fifth-Generation of Wireless Systems

Deliverable D5.2

Key Technologies for the Content Distribution Network

Version 2.0 Date: 2018/10/01

Document properties:

Grant Number:	761498
Document Number:	D5.2
Document Title:	Key Technologies for the Content Distribution Network
Editor(s):	Tim Stevens (BT)
Authors:	David Gomez-Barquero (UPV), David Navratil (NOK); Steve Appleby, Tim Stevens, Rory Turnbull (BT); Maël Boutin, Duy-Kha Chau, Nivedita Nouvel (Broadpeak); Rui Andrade Aguiar (Bundeslab); Tuan Tran (Expway); X (IRT); Baruch Altman (Live U); Waqar Zia (Nomor); Menno Bot (one2many); Francesco De Angelis (EBU)
Reviewers	Ali El Essaili (Ericsson)[1]
Contractual Date of Delivery:	2018/08/31
Dissemination level:	PU[2]
Status:	Final Version.
Version:	2.0
File Name:	5G-Xcast_D5.2_v2.0

Abstract

This deliverable builds on the use case requirements for 5G-Xcast and proposes a framework and design principles that the wider project will adopt as it investigates the flexible networking technologies required to deliver content cost effectively at scale across fixed and mobile networks. Here, we review the state of the art, identify gaps, and lay out guidelines for the implementation work packages.

Keywords

5G, framework, broadcast, design principles, multicast, unicast

Revision History

Revision	Date	Issued by	Description
1.0	2018/02/28	Tim Stevens	Interim version
2.0	2018/10/01	Tim Stevens	Final version

[1] First version (February 2018) reviewed by Richard Bradbury (BBC), Tuan Tran (EXP), Doug Williams)

[2] CO = Confidential, only members of the consortium (including the Commission Services)

PU = Public

Disclaimer

Executive Summary

This is the second of two documents that comprise D5.2, following the first, project-internal version earlier, in February 2018. Here we present the 5G-Xcast Content Delivery Framework; as such, it is not an architecture or a technical design. Instead, we discuss key attributes of a global unified content delivery framework which is capable of mixing unicast and multicast as well as fixed and mobile networks in a self-optimising way. These key attributes are:

- Infrastructure efficiency tools, such as multicast and caching, are treated as techniques for internal optimisation, not as services to be offered to the content service provider. The interface presented to content service providers should be independent of access network type.
- If possible, features will be implemented as end-point only solutions, either within the device or within the Content Delivery Network (or both), rather than requiring explicit network support.
- Interfaces between different organisations need to be kept simple. In contrast, interfaces within an organisation can be complex. Keeping the interfaces simple will limit the deterrents to take up.
- As far as possible, technology should be shared across different access network types. This includes reference architectures, APIs, service logic, media formats and protocols etc. Often differences arise simply because different standards bodies are involved and have arbitrarily chosen different approaches.
- Where multicast is used, it should be closer to the end customers, rather than deeper in the network core.
- Content Delivery Networks (CDNs) should be used for global reach.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

3GPP	3rd Generation Partnership Project
ABR	Adaptive Bitrate
ACK	Acknowledgement
ALC	Asynchronous Layered Coding
ALTSCV	Alternative Services
AOSP	Android Open Source Project
API	Application Programming Interface
ARQ	Automated Repeat Request
ATSC	Advanced Television Systems Committee
BC	Broadcast
CCAP	Converged Cable Access Platform
CCN	Content Centric Networking
CDN	Content Delivery Network
DOCSIS	Data Over Cable Service Interface Specification
DVB	Digital Video Broadcasting
EC	European Commission
eMBMS	enhanced Multimedia Broadcast/Multicast Service
FDT	File Delivery Table
FEC	Forward Error Correction
FLUTE	File Delivery over Unidirectional Transport
HEVC	High Efficiency Video Coding
HTTP/2	Hypertext Transfer Protocol 2.0
ICN	Information Centric Networking
IETF	Internet Engineering Task Force
IP	Internet Protocol
LTE	Long Term Evolution
mABR	multicast Adaptive Bitrate
MAC	Media Access Control
MBMS	Multimedia Broadcast/Multicast Services
MC	Multicast
MPEG	Moving Picture Experts Group
MPTCP	Multipath TCP
MOOD	MBMS Operation On-Demand
NACK	Negative Acknowledgement
NMS	Network Management Systems
NORM	NACK-Oriented Reliable Multicast
NRL	Navy Research Laboratory
OSS	Operations Support System
PDCP	Packet Data Convergence Protocol
PoC	Proof of Concept
OTT	over-the-top service
QoE	Quality of Experience
QoS	Quality of Service
QUIC	Quick UDP Internet Connections
RFC	Request for Comments
RLC	Radio Link Control
ROUTE	Real-time Object delivery over Unidirectional Transport
RTT	Round-Trip Time
SDI	Serial Digital Interface
SMPTE	Society of Motion Picture and Television Engineers
STB	Set-Top Box
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UE	User Equipment
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
WP	Work Package
xDSL	Digital Subscriber Line

1 Introduction

In this document we introduce the rationale for a Content Delivery Framework, describe the key technologies, and provide guidance for the more detailed technical work being carried out in Work Packages 3 & 4. Task 5.2's job is to present a framework and principles. The task does not set out to describe an architecture, nor a technical design nor to make standards recommendation. The framework and principles developed in Task 5.2 and reported here should be repeated across the project as it will provide the underlying rationale that drives subsequent detailed technical work. For example, the technology gap analysis work that originated in Task 5.2 is reported project-wide in WP2.

Here we take the high-level ideas expressed in D5.1 "Content Delivery Vision" [1] and discuss how the vision of simple network interfaces, unicast delivery and treating network type and transport formats as internal optimisations lead us towards keeping the intelligence out of the core networks where possible. We take it as a given that technologies and services will evolve rapidly and assert that our framework and principles should support this. To do so, given the reality of commercial factors such as the relationship between network operators and CDN operators, it makes sense to keep intelligence, QoS and the "hard stuff" close to the network edge, or to confine them to the end-user devices.

1.1 Release 1 and Release 2

D5.2 has two phases: Month 9 (February 2018) and Month 15 (August 2018). This issue of the document forms the final report, with the task being complete as far as 5G-Xcast is concerned. D5.2 should be judged a success if the thinking that arises from writing it is reflected in improved quality and better scope throughout the project as a whole.

2 Media Delivery today

Networks exist to deliver content. Such content may be data, voice or video. It may be transmitted between pairs of devices, or in huge broadcast networks and it may be at very high or very low data rates. Whilst 5.2 must be mindful of all the 5G-Xcast requirements [2] (expressed in WP2), we are most concerned with the Media & Entertainment use cases, and from an application-level view, rather than a transport-level one. Figure 1 shows some of the common media paths and formats used for video production and distribution, although 5G-Xcast is primarily concerned with distribution.

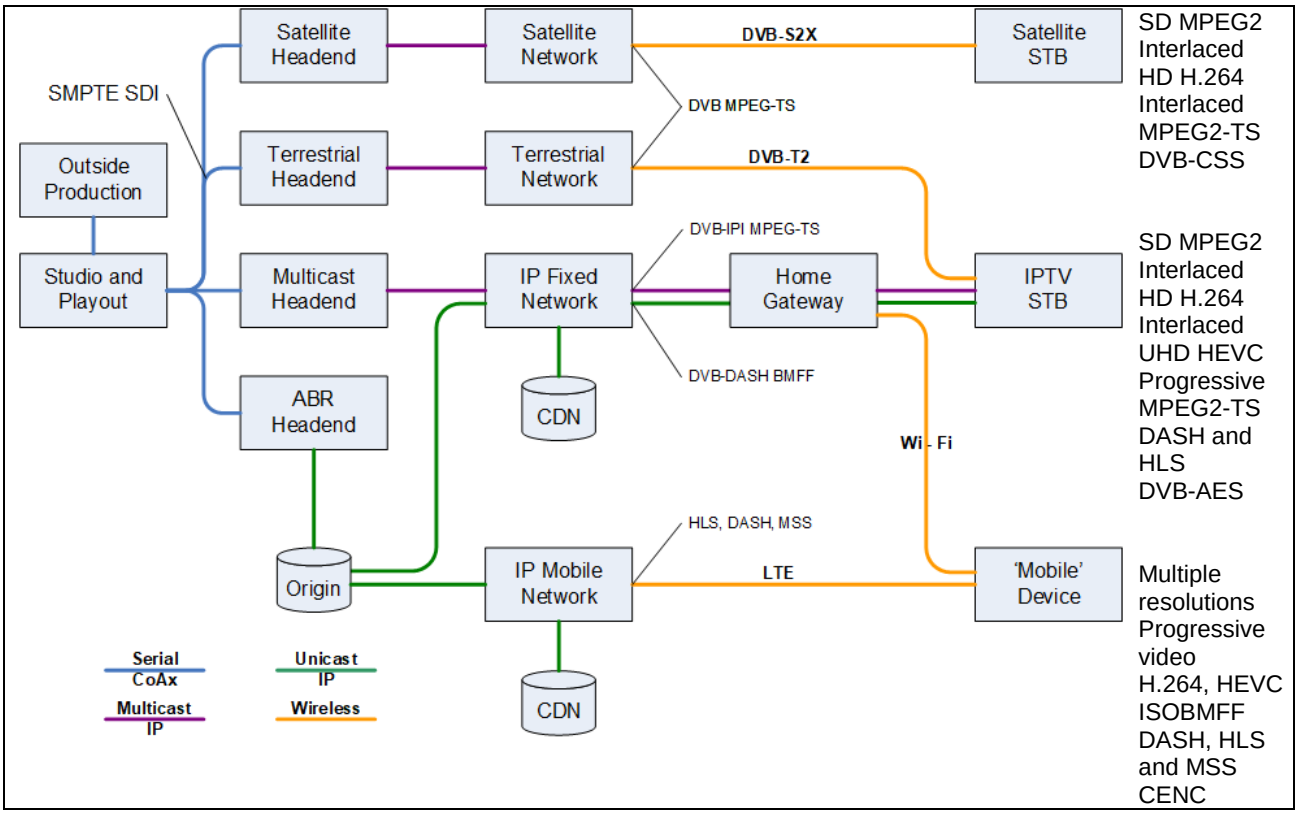

Figure 1 Typical media flows across different networks and devices

Note that Figure 1 is not exhaustive, and is a simplification. For example, we could also argue that the output from 'IP Mobile Network' could be an input to 'Satellite STB', since it could be used to augment a poor-quality satellite reception. Such hybrid bonding is not yet common. However, the current trend for convergence of both service and technology may drive an increase in the uptake of such hybrid solutions.

2.1 Key Goals

Content is delivered at scale today using general-purpose protocols. In the fixed network, unicast HTTP streaming dominates, and network operators rely on CDN providers to manage large-scale delivery of content. IP multicast is also used by some network operators, but since a reliable and scalable multicast estate requires tight control of the ingestion nodes, multicast remains under the complete control of the network operator, and does not generally cross organisational boundaries.

Turning to mobile, unicast HTTP streaming is reliable and allows 4G customers to consume high-quality video service on smart devices. Until recently (LTE Broadcast, eMBMS) broadcast mode had not been widely adopted in LTE, however as of September 2018, commercial services are being launched, with forty-one operators having invested

in eMBMS pilots and trials, and five having now deployed eMBMS or launched some sort of commercial service using eMBMS[3]. Further background information on how 5G offers an opportunity to improve on the 4G-LTE Point to Multipoint broadcast technology is presented by several 5G-Xcast authors in [3].

A key goal of 5G-Xcast is to develop techniques and standards to allow broadcast mode over mobile to flourish.

Equally important is the complex web of commercial arrangements between chipsets, devices, network operators, CDN providers and content owners.

Wide scale adoption of fixed/mobile broadcast convergence is not therefore just a technical and standards issue: the stage must be set for the commercial and economic arrangements to work too.

This work package aims to steer 5G-Xcast towards outputs that optimise both technical and commercial conditions for the uptake of broadcast in 5G. To be successful, we need to move from platform-specific solutions to an ecosystem that allows content and services to flow through networks in the most appropriate way, using a combination of fixed and mobile networks, and using unicast, multicast and broadcast delivery as appropriate.

The detail of the technology may be complex, however the 5G-Xcast framework must keep the barriers to adoption as low as possible. This implies making the inter-organisational interfaces as simple as possible, and as far as practical, treating the use of specific technologies as an internal (system) optimisation, opaque to the content providers and end users.

2.2 LTE-B, and how 5G-Xcast can build on it

Deliverable D4.1 covers technical details of the eMBMS Release 14, and its limitations. We do not reproduce this information here although we note that, as an evolving standard, the technical situation will no doubt be refined. Of interest to this work package are the less-technological issues that may be limiting the uptake of LTE-B. We hope that the project overall will combine the technological and wider factors and thus identify ways to increase LTE Broadcast uptake, based on the work done with 4G.

2.2.1 Disruptive technology: wider use cases

Until relatively recently, broadcast technology tended to operate in integrated domains. Conventional television services, for example, may be supplied via competing technologies including digital broadcast terrestrial and satellite technologies. Cable systems such as DOCSIS are also common and, increasingly, Ethernet-based IP is being used for TV delivery too. Delivery technologies, and also the packaging of the media, tend to be specific, and changing from one to another may require the consumer to purchase new equipment (satellite dish and decoder to replace aerial and decoder for example).

However, the trend now is towards an Internet-centric view, where generic technologies and IP-based protocols are moving us to a position where the same content may be consumed on almost any device. There may still be domain-specific media encoding or content protection in this case, but the emerging principle is of ubiquity of delivery. This of course has disruptive implications for broadcasters, content providers and network operators. Unsurprisingly, there is considerable inertia due to the technology, legal and commercial relationships that are embedded in the existing solutions. This has probably

[3] https://gsacom.com/paper/lte-broadcast-embms-market-update-2/

not helped the rapid adoption of the new broadcast-mode opportunities by the older players.

Turning specifically to mobile, discussion during the development of LTE-B focused on video distribution to specific situations such as sports stadia for broadcast scenarios. This is a sensible use case since it clearly involves large numbers of people wanting to receive the same content whilst conveniently being located in close proximity. However, it is still somewhat niche, given that large numbers of smartphone users don't spend significant amounts of time at live mass-audience events, neither is there a clear demand from this audience who have paid to see a live event, to spend their time looking at a 5 inch screen.

However, with the growth of other sectors such as Internet of Things (IoT), connected vehicles, public warning systems etc., there may be sufficient commercial opportunity for providers to deploy LTE-B to address these applications. The pace of technological change in vehicles means that new services are emerging, and the connected vehicle scene may be very different in five years' time. Similarly, the huge growth in static IoT devices, many with tight cost or power constraints, may make them natural consumers of 5G services.

Returning to broadcast, mobile bandwidth is expensive relative to fixed, and in a world of increasing demand, it makes financial sense to move to broadcast where possible in the mobile case, even where this may not be true in the fixed situation.

2.2.2 Devices and Licensing

Apple currently does not enable LTE-B on its iPhones (although the underlying hardware may support it). Multiple Android devices have already supported LTE-B. Google have added eMBMS support to the AOSP (Android Open Source Project), with release 8.1, which may help to encourage take-up. The adoption on iOS devices would make the use of LTE-B more beneficial, if Apple choses to enable it.

2.3 Business Models

CDN operators charge according to the volume of bytes served from their edge nodes on the networks. The use of multicast at the edge will reduce the number of bytes delivered from the (unicast) CDN, compared with a conventional unicast to the edge delivery. This may undermine CDN revenues so must be given due consideration.

Also, in order to support HTTPS, CDN operators will need to be in agreement as serving devices need to have the appropriate certificates. In the multicast case, these devices will be in the NSP domain. This will require either CDN operator certificates to be installed on NSP devices, or the CDN operator actively redirecting to the NSP domain.

On a fixed network, multicast usually operates at a constant bit rate over a guaranteed bitrate connection. Because of this, content service providers may be persuaded that they are receiving a service that offers a guaranteed quality and that should, therefore improve customer satisfaction. On the mobile network, it is possible to use a Single Frequency Network, which could achieve better coverage, avoiding edge-of-cell performance issues and similarly offer the promise of a service with more reliable characteristics and a better overall quality of service for end users.

There are clearly mutual benefits to be had but in any case, a commercial and technical relationship with the CDN operator will be required. Such relationships already exist of course, but an agreement would need to be reached on the use of multicast, which is somewhat disruptive to current practice.

Addressing these commercial and organisational issues is outside the scope of 5G-Xcast. However, we need to define a compelling case for adopting our framework proposals if we wish to see them adopted.

3 Framework Design Principles

Section 2 argues that the slow adoption of the broadcast mode in LTE is due to issues that are both commercial and technical. For adoption, it is not sufficient to specify an elegant engineering solution to unicast/broadcast delivery; the vendors, operators and content providers must see a benefit, and if their current business practices may be disrupted, that benefit will need to very clear. We now introduce a set of design principles that should help guide the engineering decisions so that as well as designing good technology, the inter-business relationships are also considered.

We have agreed to adhere to the following design principles across all work packages as far as realistically possible. These are high-level principles, and are not intended to be detailed design instructions! Our goal is to shape the thinking so that, for example, 5G-Xcast doesn't require organisations to tear up established business practices, or be forced to cede control of their content or traffic; we need to develop solutions that encourage collaboration and flexibility. The principles:

- Multicast/broadcast should be treated as an internal network optimisation issue.
    - There should be a mode where the multicast/broadcast transmission choice is hidden and not necessarily sold as a service in its own right[4]. Some use cases such as V2X and PWS may however benefit from a direct multicast delivery: this needs further investigation (This does not exclude exposing MC/BC as a service, but this is not the actual focus of our work.)
    - Combine CDN for global reach with multicast/broadcast for edge optimisation.
- Handover to and from the multicast/broadcast network segment is unicast IP
    - All content services are treated as OTT services, because there is a benefit to operators if they can treat all content as if it was OTT, and not have to deal with deployment of QoS across the whole architecture. The infrastructure should intrinsically allow this.
    - Minimal impact on content service providers, CDN operators etc.
- Keep interfaces simple across organisational boundaries.
    - E.g. cross-organisational resource reservation needs trust/authentication, billing, clear 'product' built around resource value etc., although this creates a barrier to deployment.
    - In such cases, prefer autonomous resource allocation (with no exposed API at the application layer; instead allow the network to manage the routes & resources itself.), even if less efficient.
- Where possible, features to be implemented at the endpoints only, rather than within the network; with application-layer intelligence preferred over network signalling.
    - Even at the expense of efficiency.
- Consider vertically-integrated versus distributed ownership, and implications of moving the boundaries.
- Non-specific design. Meaning, a design that enables all foreseen applications as considered in WP2, in the benchmarking, in "standard" industry practices and in network operators' deployments, unless one of these is too far beyond the mainstream.
- Specify the mandatory and the optional features/functions/entities.

[4] Deliverable D5.1 makes thecase for this. It's proven difficult to commercialise multicast as a service that people will buy is it comprises islands of technology in a unicast internet. This makes it hard to optimise the mix when the parts are all under integrated control (which would require cross-organisational resource management).

At the WP5 level, these general principles will not be prescriptive, and the other technical work packages that make the specific recommendations will adopt and specialise them.

CDNs became an essential technology for building content delivery services. The role and offered benefits of CDN in future content delivery systems is carefully considered. A CDN is essentially a network of caches, and the users are served from the CDN edge nodes 'closest' to them (using some appropriate cost metric, service availability metric etc.). However, to reach the users, the data must flow through the local distribution provided by the network service provider. The CDN edge nodes may be inside the NSP's network. A CDN effectively:

- improves scalability of the system and adopts to user demand for content by employing multiple distributed servers and thus avoiding computing and transport bottlenecks of a single server;
- reduces traffic in core networks, interconnection and peering links;
- improves service availability; and
- lowers delays as content is served from a near location.

The slow adoption of multicast and broadcast does not apply only to LTE but also to the Internet. Multicast does not readily traverse the open Internet, or across operator boundaries. Multicast can of course be unicast-tunnelled using techniques such as Automatic Multicast Tunnelling [4]. AMT provides a method for UDP-encapsulated tunnelling of multicast data over unicast-only networks from an AMT relay located in a native multicast network to an AMT gateway located in an isolated site or a host itself. The AMT gateway discovers AMT relays using any cast IP address. AMT offers the following features:

- simpler tunnel establishment and management in comparison to other tunnelling solutions which require manual provisioning and management such as GRE tunnels;
- resiliency achieved by the deployment of multiple AMT relays and automatic discovery using any-cast IP address;
- efficiency in terms of link utilization and sever load hosting content.

It is interesting to note that the take up of AMT for general services has been slow.

CDNs and AMT take different approaches to address the problem of content delivery. AMT aims at enabling content distribution from one or few central locations on global scale over the Internet. In the case of CDNs, the content is distributed to the CDN edge nodes from where it is served to users. These two architecture approaches could provide similar latency of multicast data for example considering a live content production and delivery. However, if multicast data is not received correctly then unicast repair mechanisms can benefit from lower delays provided by CDNs. It therefore makes sense to propose a framework that exploits the global scale of CDNs, and uses multicast as a delivery optimisation option for the customers within an operator's domain, without assuming the support of native multicast across the wider Internet.

Given the benefits and commercial success of content delivery networks, the framework also considers the multicast networks to be closer to the edge when we have a global infrastructure, compared with the more centralised multicast estate used where there is a single operator (Figure 2).

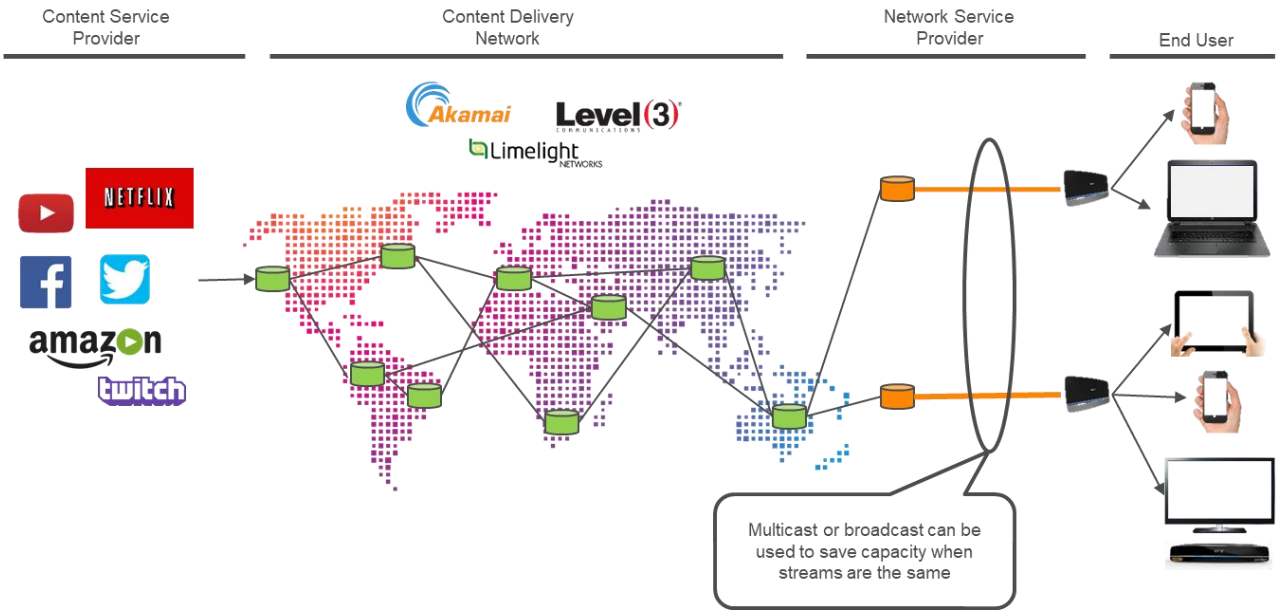

Figure 2 CDN for global reach, dynamic selection of multicast at the edge

We therefore have alternatives of unicast, multicast and broadcast, with fixed and mobile networks closer to the customers.

4 State of the Art and Gap analysis

WP5 generated an initial benchmark of current technology, subsequently extended to include the WP3 and WP4 perspectives. We have done this by taking the requirements generated in WP2[5] (comprising the three main use case areas of Media & Entertainment, Public Warning and Internet of Things with their individual requirements: a total of roughly 65 individual short requirement statements), ranked them by (approximate) priority, and indicated how they relate to each WP, and what technology might address them. This work has allowed us to show that our 5G-Xcast requirements have been addressed at the next investigative stage. Further information is available through the WP2 deliverables.

4.1 CableLabs as an example Framework

The CableLabs reference architecture[6] uses the Cable-based DOCSIS local network, although the design goals are relevant to other distribution technologies that support bidirectional traffic over reliable connections, and the principles could be applied to a mixed 5G/broadband network. The goal of the specification is that the client always receives unicast content via HTTP(S), however the content may traverse the network as either unicast or multicast, with the decision being an internal optimisation. The MC receiver can be a setup box or gateway. NORM (RFC 5740) is used as the encapsulation format for multicast transmission.

A slightly simplified version of the reference is shown in Figure 3, with the key elements in green and the control interfaces in red.

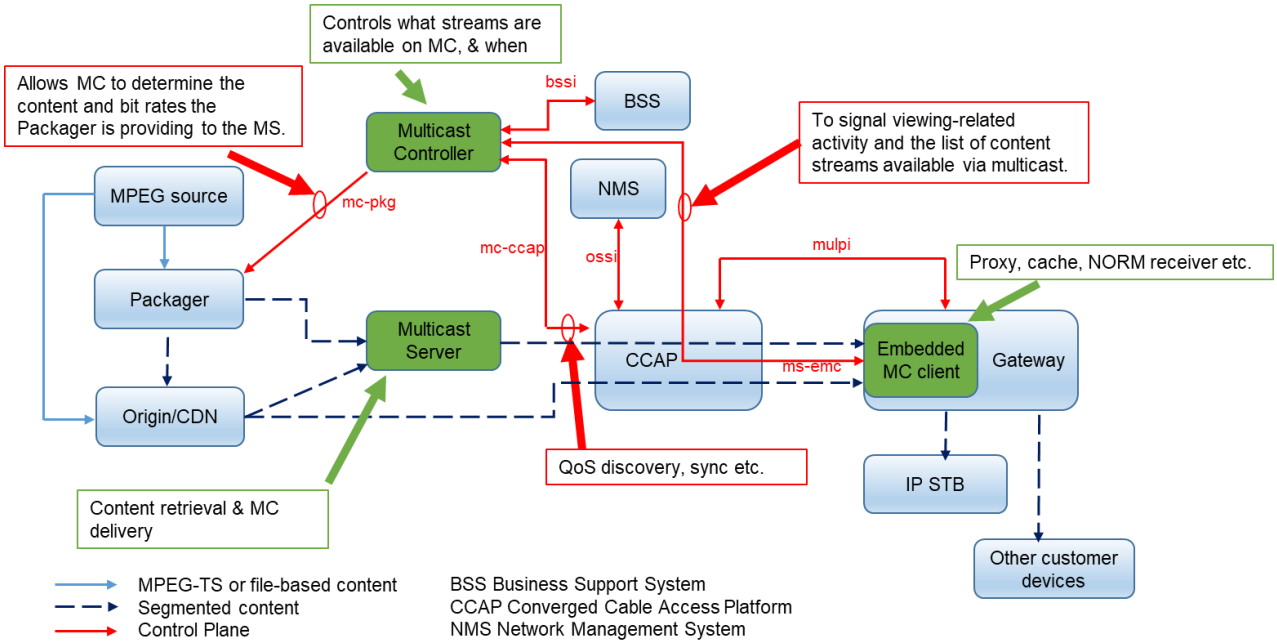

Figure 3 Simplified CableLabs Reference Architecture

[5] 5G-Xcast_WP2_037_Requirements_From_WP3_WP4_WP5.XLSX

[6] IP Multicast Adaptive Bit Rate Architecture Technical Report OC-TR-IP-MULTI-ARCH-C01-161026,
IP Multicast Controller-Client Interface Specification OC-SP-MC-EMCI-I02-160923,
IP Multicast Controller-Server Interface Specification OC-SP-MC-MSI-C01-161026, and
IP Multicast Server-Client Interface Specification OC-SP-MS-EMCI-C01-161026

Multicast traffic is delivered at a fixed, constant rate – it is essentially *pushed* to the multicast clients. However, unicast clients will generally *pull* content: with the common formats such as ISO-BMFF/DASH or MPEG2-TS they parse the manifest, and request the referenced segments by HTTP, and expect the segments to be delivered faster than the playout rate. Clients maintain internal buffers, allowing throughput variation to be smoothed out. This presents a problem when content may be routed via either type of network, since the multicast path cannot supply content faster than the 'natural' delivery rate. To avoid this problem, CableLabs recommends that the embedded MC client in the Residential Gateway modifies the manifest before supplying it to the client on the end device, commonly removing the reference to the final segment. The embedded MC client therefore has more knowledge of the stream than the end client, and thus has some headroom to decide whether to fetch the 'missing' segment over unicast or multicast.

4.1.1 Packet loss

CableLabs includes provision for endpoints to detect and request missing packets: the specification recommends FEC but supports reactive repair via retransmission requests. For FEC with NORM, the fec_payload_id value is used to allow integrity checking of each data transfer and have packet loss or reordering detected and handled reliably.

NORM also supports a sophisticated scheme of allowing clients to send Negative Acknowledgements (NACKs) back to the multicast server. The server can collate these responses from multiple clients and send the best set of new FEC packets to fill the gaps.

Alternatively, clients can also use out-of-band methods to repair packets, for example by using HTTP range requests to fetch the gaps from the CDN.

4.1.2 Cablelabs Suggested best practice

- Utilise the NACK Oriented Reliable Multicast (NORM) protocol as the multicast transport protocol.
- Use Source-Specific Multicast in addressing (Source address, Group Address)
- NORM FEC is a recommended practice for reducing repair traffic.
- Utilise unicast repair when errors occur in data delivered via NORM (don't use NACKs)
- The Gateway should utilise HTTP Range-Requests for repairing missing video segment data.
- The Multicast Server should utilise NORM INFO messaging to deliver HTTP header info associated with a given video segment such that the Gateway can reassemble the full HTTP response from the Origin/CDN for the Player.
- Broadcast a Multicast Channel Map providing the mapping between a URI and the appropriate multicast address
- Stream output should be paced to transmit chunks over the chunk length duration
- The system should have the capability of modifying or managing manifests to allow the Gateway's Embedded Multicast Cache to stay at least one segment ahead of a Player's requests
- The Multicast Controller should know that a Gateway is "Tuned but Not Viewing" so that it can determine when to potentially terminate the multicast of a given stream.
- The Gateway shall function as a transparent proxy intercepting requests for appropriate URLs[7].

[7] This has implications for HTTPS handling.

4.2 Why Cablelabs is not a solution here

We have introduced the Cablelabs work as an example of a good framework and guidelines. However, it would not be appropriate to simply adopt Cablelabs for 5G-Xcast. The main reasons for this are:

- Cablelabs takes advantage of a relatively tightly integrated ecosystem, to deliver multicast-ABR. As such, there are many data- and control-plane interfaces (more than are shown in Figure 3), and these complicate the business logic and relationships. For example, the mc-pkg interface allows the media packager to communicate with the multicast controller. Whilst this provides useful information to the system, it implies a commercial arrangement that may not exist in the open world of 5G-Xcast.
- Cablelabs assumes a uniform access network. The specification relates to DOCSIS cable delivery (although it could in principle be extended to other access network types), where access are managed using frequency division multiplexing and specific modulation schemes. 5G-Xcast, in contrast, will have a mix of IP-based wired and Wi-Fi, unicast and multicast plus 3GPP radio delivery, in broadcast or unicast modes.
- 5G-Xcast also needs to interface with multiple Content Providers, Network Operators, CDN Operators and mobile networks, requiring a more open, flexible approach: we need to offer solutions that cross technical and commercial boundaries without forcing these incumbents to radically change their current practice.

For these reasons, the Cablelabs spec does not meet our needs.

4.3 DVB

The DVB Project is conducting relevant studies in this area, and we summarise the work in annexes A.6 (file casting) and A.7 (multicast-ABR). Both of these address transporting object-based content in a non-unicast carrier; using broadcast or IP multicast respectively. The m-ABR work in particular addresses some areas that are common to 5G-Xcast, and has defined a reference architecture (from which we have borrowed the terms “**Function X**” and “**Function Y**” that we use in this document, to refer to multicast server and client-like functions respectively). However, whilst the DVB’s m-ABR task force is mindful of mobile networks, transporting media over them is not core to its work.

Also, both activities are currently too early in their progress to be directly able to influence 5G-Xcast. However, we have project members common to both 5G-Xcast and the relevant DVB working groups, and we are also feeding 5G-Xcast thinking into the DVB discussions.

4.4 Object-Based Broadcast in ATSC 3.0 ROUTE/DASH

Object-based broadcasting partitions the service content into objects to be delivered to a receiver end, where it can then be assembled according to the receiver device features. With these properties, object-based broadcasting has been an enabler of flexible delivery of media content and interactive user experience that is highly responsive to individual needs [5].

One of the practical deployments of such broadcasting is the DASH-based ROUTE delivery in ATSC 3.0 systems [6]. The ROUTE/DASH broadcasting method is designed to deliver DASH-formatted streaming content over ROUTE in the form of objects to a large number of broadcast receivers. This method is based on the DASH-IF Interoperability point for ATSC 3.0 and consequently MPEG DASH. ROUTE is used for the delivery of DASH segments. Additionally, ATSC 3.0 integrates DASH-based

streaming over HTTP/TCP for the broadband transmission. These two methods introduce the unified broadband and broadcast approach, in which unicast routes can be used for combining additional service features to broadcast service, e.g. availability of subtitle support for different languages through unicast for a broadcast video/audio service. As a result, this enables flexible service composition, which is a key goal for object-based broadcasting. The above mentioned DASH-based unified broadband-broadcast protocol stack of ATSC 3.0 is shown in Figure 4 [7].

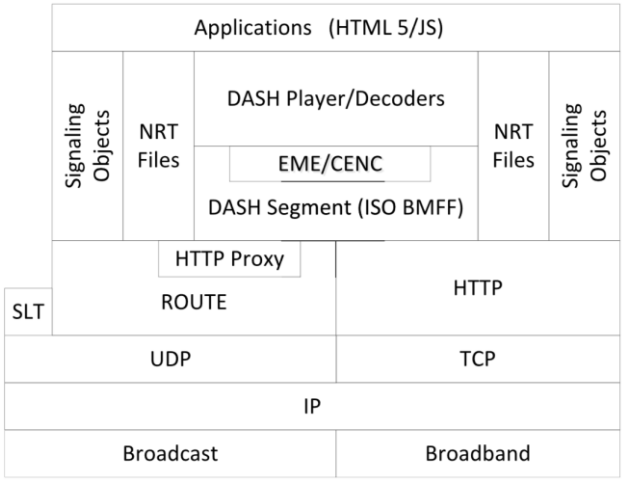

Figure 4: Unified broadcast and broadband protocol stack.

With ROUTE/DASH, each media item is fragmented into objects to be delivered through predefined and pre-declared LCT channel(s) optionally along with the metadata that is used by the client ROUTE application for the assembly for playback in a timely manner. The metadata in this case is called Extended File Delivery Table (EFDT) and contains information such as the content location, content size, media type, etc. By parsing the received EFDT, the client puts together the related received objects and generates the presentable DASH-formatted data to be consumed.

As stated in Section A.3, EFDT can be sent either statically in the service layer signalling phase or dynamically in the same LCT channel as the source flow. In the latter case, the EFDT data can be not only modified and/or updated depending on the changes to the delivered service but also, transmitted either in-band with the delivery object(s) in the form of a compound object or as LCT extension headers in the packet headers enabling fast channel zapping time and low initial playback delay. Therefore, this system provides a multicast and broadcast delivery solution combining both the aforementioned advantages of the object-based broadcasting and the advanced features supported by ROUTE signalling and delivery.

A representative schema for this model is depicted in Figure 5 below. In simple terms, the object(s) associated with a specific discrete time slot that is specified by the physical layer scheduler is transmitted via the transport buffer, TB_n. These data blocks are temporarily and briefly stored in the *ROUTE output buffer* before consumed by the client application. The EB_n buffer is associated with the DASH segment handler functionality, which bundles the related objects together in a way that they would represent a meaningful presentation for the decoder and conveys this to decoders according to the specified presentation timeline.

Since the repair flow is generated on object-basis rather than packet-basis, the same logic also applies to repair flow. The only difference in this case is that the “meaningful presentation for the decoder” comprises both source and repair data blocks for the corresponding playback timeline.

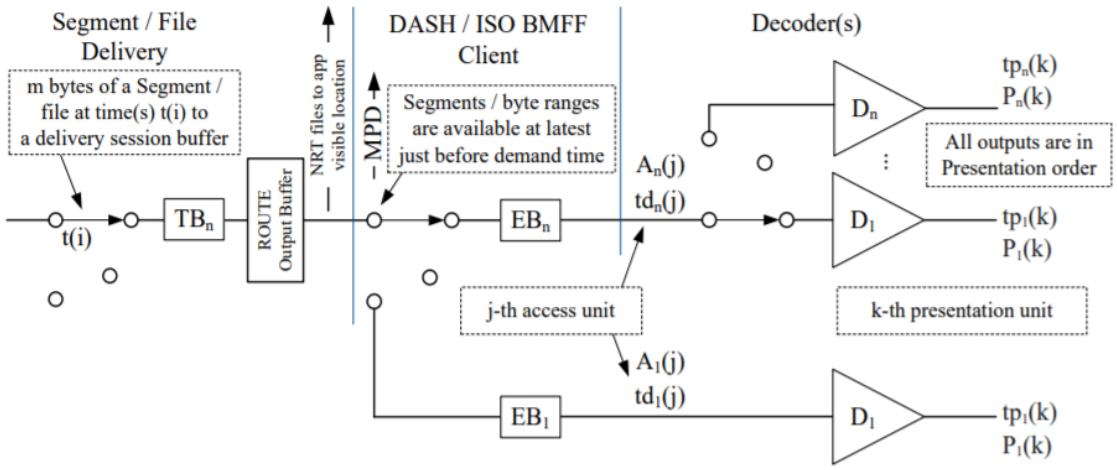

Figure 5: ROUTE/DASH system model.

4.5 Streaming Video Alliance

This section provides a summary of the activities of the Streaming Video Alliance (SVA), with a focus on the OpenCaching (OC) initiative[8]. The Streaming Video Alliance is a group of companies working in relation with the streaming video value chain. The SVA is not a standardisation body but it aims to contribute to standards with the outputs of the working groups. Its primary focus is on developing best practices, guidelines, technical specifications, and functional requirements that address critical challenges in the online video industry. The technical work is arranged according to Topic Area and then implemented through specific Working Groups.

The SVA use the concept of the Open Cache Node (OCN) as a building block for CDNs, and extend the IETF work defining a Content Delivery Network Interconnection (CDNI) and many of the use cases identified in RFC 6770.

The elements of interest here are shown simply in Figure 6, and include:

- OCN, which is essentially a universal multi-tenant cache function deployed and owned by the service provider in close proximity to the users. A typical service provider realm may employ multiple (100s-1000s) of OCNs. The primary goal of an OCN is to deliver content to users within the service provider realm while relaying logging and billing information to upstream CDNs that delegated traffic to it.
- CDN Open Cache Controllers (OCC) – a control function used by delegating CDN enabling the CDN to gain access to open cache resources inside service provider networks. The CDN OCC communicates with multiple service provider open cache controllers aggregating global open caching data by communicating with OCRCs.
- Service Provider Open Cache Controller (SP OCC) – a control function used by a service provider with an open caching deployment interworking via API with CDN open cache controllers. The SP OCC represents the entire service provider open cache realm towards the CDN OCCs, tracking all OCNs' location, status, capabilities and subscriber mapping while acting as an OCN registrar.

[8] https://www.streamingvideoalliance.org/technical-work/working-groups/open-caching/

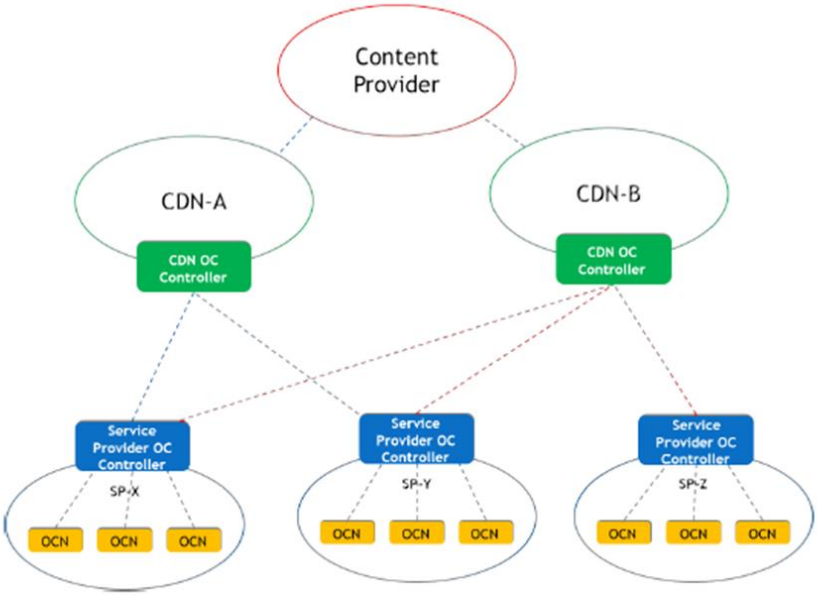

Figure 6 Open Cache Node relationships

Functional requirements are defined for the Open Cache elements such as the nodes and controllers. As with other cache/CDN solutions, these requirements cover areas such as content handling, acquisition, validation, security, logging, monitoring etc. plus service management, orchestration and control.

The OC architecture enables CDNs to delegate content requests to Open Caches located at the edge of the SP network. Once delegated, client requests arrive at an OCN, the OCN delivers the content as a proxy, while at the same time it caches it for delivery of later requests. Thus, OCNs are required to store and deliver content items delegated by the CDN on behalf of the CDN's customers (the content providers).

The OC architecture requires the system to support a set of content management operations, allowing the CDN to gain control over the content it is delegating to the SP OC System.

In a traditional content delivery scenario, without OC, the CDN provides the CP with a set of interfaces allowing the CP to operate on its content, which is served by the CDN. In OC scenarios, the OC system must support the same content operations, such that the content operations offered by the CDN to its CP customer are maintained.

The OC content management interface enables the CDN to instruct the ISP OCC to perform content-related operations such as pre-positioning, revalidation/invalidation and deletion.

4.5.1 Request Routing

Request routing and he associated redirection of the requests/responses is an important topic for 5G-Xcast, and so we briefly present the OC approach here. Figure 7 and Figure 8 show the simple redirect and query-redirect cases.

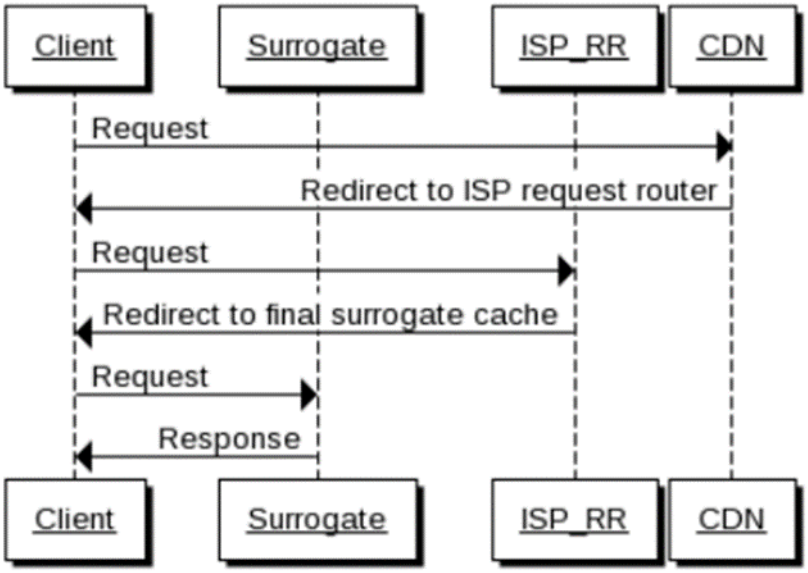

Figure 7 Iterative Request Routing

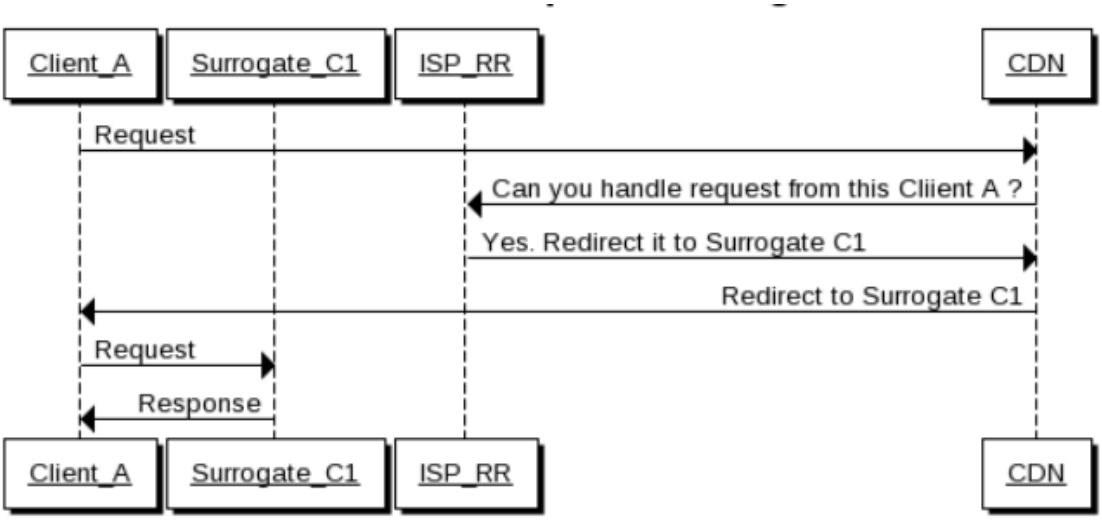

Figure 8 Recursive Request Routing

There are several approaches to redirection, and we mention this further in section 6.

4.5.2 Delegation Scenarios

A CDN can delegate traffic to an Open Caching system once the following conditions are met:

- The CDN has received the OC system capabilities and validated that they match its needs.
- The CDN has received the OC system footprint and has validated that it has coverage for the subscribers it whose requests it wishes to delegate.
- The CDN has advertised its metadata to the OC system and received acknowledgement from the OC that the metadata was received and processed.
- The OC system capacity advertisement indicates there is free capacity.

The CDN must stop delegating traffic to the OC system in one the following events:

- An OC system capabilities update indicates that there is no longer support for a required capability. For example, support for HTTPS is no longer available.
- An OC system footprint update indicates that there is no longer coverage for the relevant subscriber zone.
- The OC system capacity advertisement indicates there is no more free capacity.

Request routing flows are illustrated below:

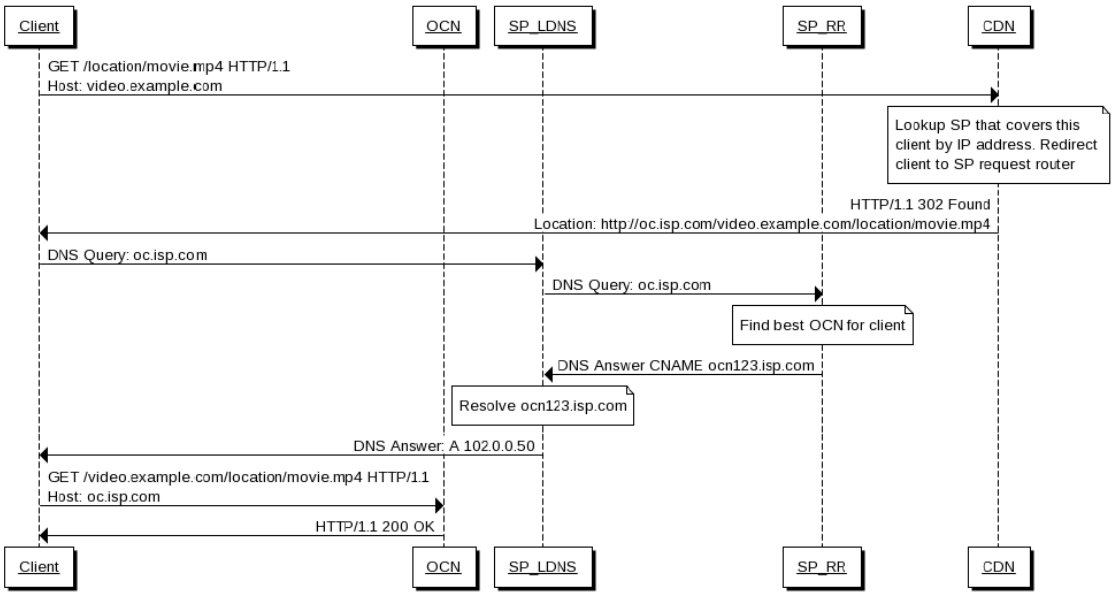

Figure 9 HTTP 302 Redirect

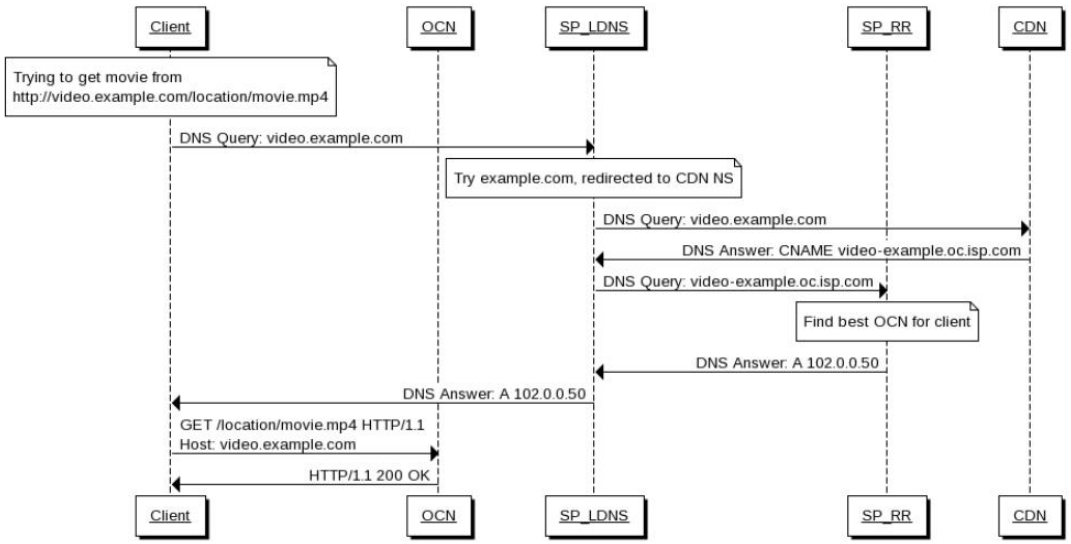

Figure 10 DNS CNAME Redirect

4.5.3 HTTPS

The Open Caching system is required to be as secure as the CDN that is delegating traffic to it. HTTP delivery security is achieved by using TLS as the underlining transport for HTTPS. The OCN is required to support standard X.509 certificates. Certificates can

be custom for specific domain or shared using SAN (Subject Alternative Name), depending on the requirements of the delegating CDN or its CP customers.

As certificates are issued to a specific domain name, there are significant differences between the cases of HTTP request routing, where the redirect domain name is of the SP OC system, and DNS request routing, in which case the redirect domain is the same as of the delegating CDN.

4.5.4 Observations

- The objective of the Open Caching initiative is to have network operators implementing standard APIs to handle the delivery of content provided through CDN services. CDN services are the entity who sign a deal with content providers but whose responsibility in the content delivery stops at the gates of operators.
- The standard is very close to IETF CDNi about CDN federation that was active in 2010.
- SVA seems to be driven by Qwilt, who provided transparent caching solutions to network operators. Transparent caching became inefficient with the adoption of HTTPS, made mandatory by big players such as Google and Apple
- The logic behind the SVA approach is close to the transparent caching concept, to which it adds a contractual formalism between content providers, CDN services and operators.
- US network operators appear to be the only active participants in the consortium
- An important part of the work to be done to comply with the recommendations is at the level of the operators' own networks, and hence would require involvement with their CDN solution providers.
- The interaction and collaboration between CDNs and operators implies the development of some specific protocols and requires the exchange of an important quantity of additional information that will be used for cache selection and for billing.
- Latency in the system will probably be added if these new protocols are used for system discovery, since a redirect is added to the standard process.
- The business models, and hence billing relationships appear uncertain, which is a major drawback for deployment.
- Regarding Service Level Agreements, it may be hard to determine the responsibilities in case of issues.

5 Content Delivery Framework Overview

Section 2 introduced the commercial background to LTE, and discussed some of the reasons why it has, to date, not been widely adopted for media broadcast, despite being a successful technology for reliable high-speed unicast delivery: the commercial relationships are vital.

Section 3 proposed the design principles that (we hope) a successful 5G project should embrace, since by doing so, the commercials as well as the technical challenges will be addressed. Section 4 summarised a deployed framework, and discussed why we can't simply adopt CableLabs for 5G-Xcast. We also introduced other approaches such as AMT and Open Caching as background and potentially-helpful technologies. Our principles should shape all the 5G-Xcast thinking, and we now introduce a framework that embodies them.

5.1 Functional Entities

Figure 11 shows a simplified representation of the components and media flows for 5G-Xcast. The aim is to show how content can flow over mixed network types comprising fixed and mobile, and unicast with multicast and broadcast connection types. As already stated, this is not intended to be a 'design', and nor is it exhaustive: we arrived at it as a reasonable way to express the kinds of issues that need to be considered across 5G-Xcast as a whole.

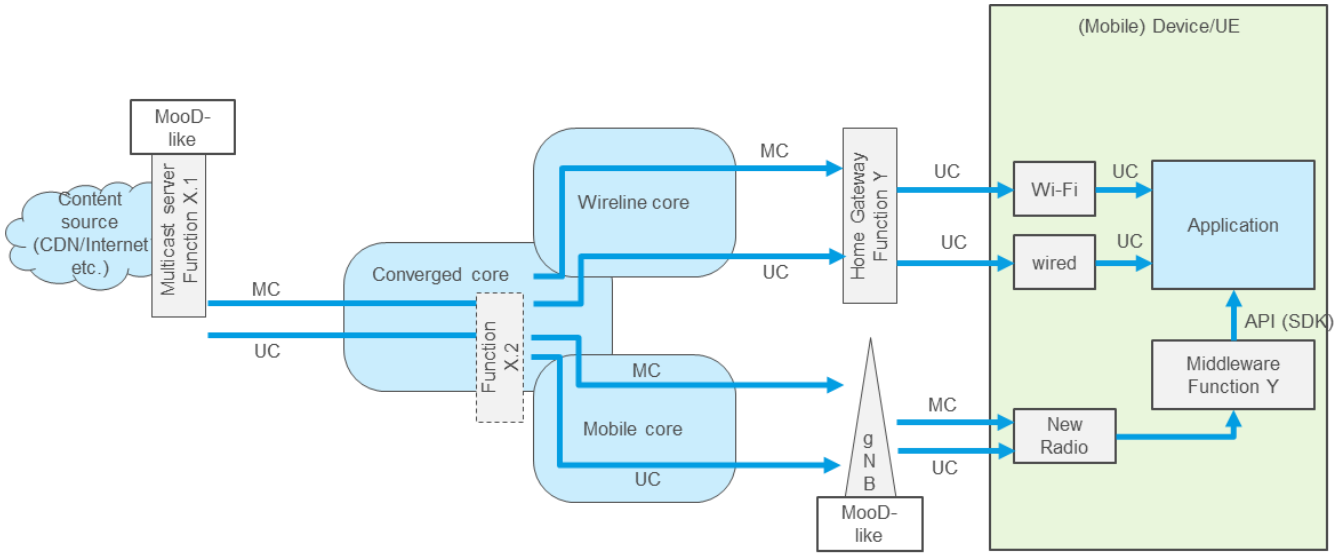

Figure 11 a possible 5G-Xcast framework

We acknowledge that there are alternative arrangements for many of these logical blocks, and have shown this for example with the two-part multicast Function X head-end; the diagram intends to show that these functions need to be considered, but is not prescribing how they are deployed. Similarly, the Home Gateway might also be capable of receiving radio, but we have kept the logic separate here.

We use the term 'converged core' to indicate that both user plane and control plane data and signalling are present, allowing both fixed and mobile [8] delivery paths to be used [9].

Figure 12 superimposes the relationships between the main technical work packages 3 and 4, and the tasks in work package 5. The aim is for implementation issues and

technical decisions to be delegated to the other tasks, and it is the responsibility of T5.2 to see that these are addressed, even if at this stage not all can be answered.

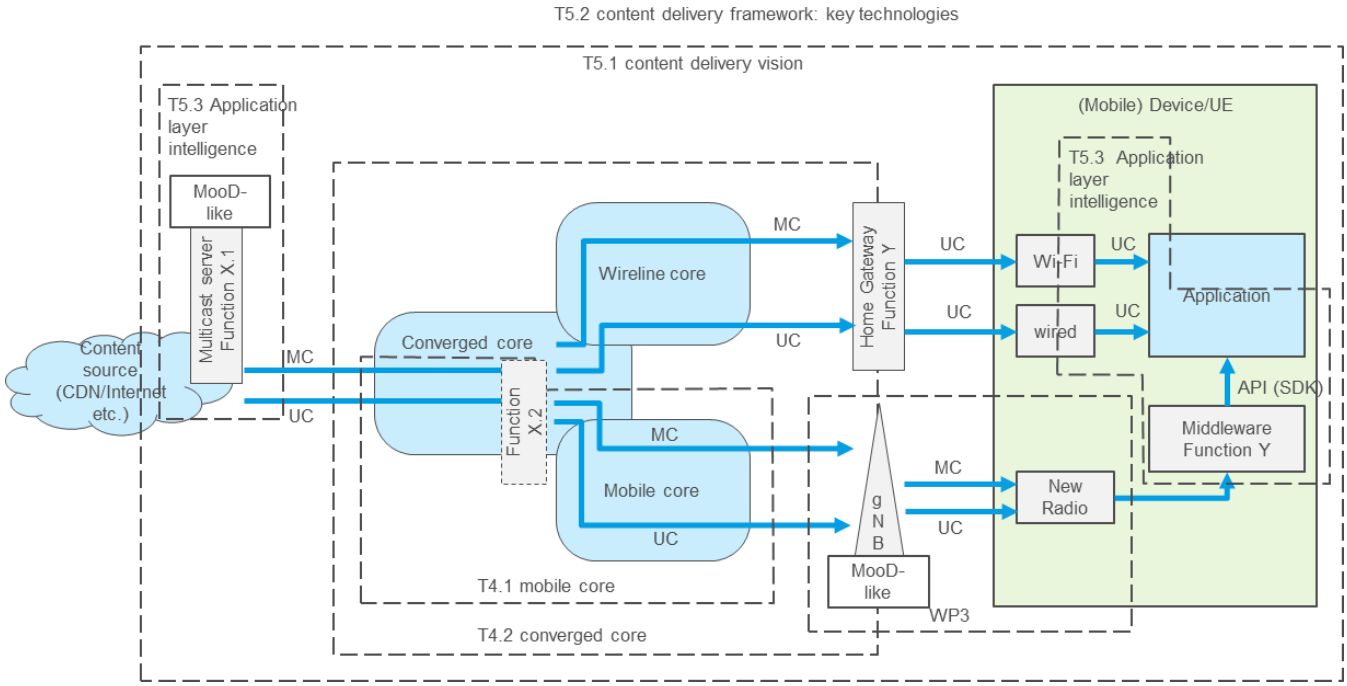

Figure 12 how the tasks relate to the Framework

Not all the tasks are shown in Figure 12. For example, T4.3 convers session management and control. These are an essential set of functions that span the project as a whole, however representing them on a diagram such as this does not really convey their interaction with the project as a whole, and makes the diagram less simple to interpret. They are of course vital functions none the less.

5.2 Functions X and Y

We introduce two logical functions X and Y, to handle the multicast transport of unicast data. We are referring to these by the letters X (multicast server end) and Y (multicast termination end) rather than more descriptive names because we want to be technology-agnostic as far as possible when discussing the framework. Terms such as “head end”, “server” etc. mean different things in different contexts, and the labels X and Y are therefore useful placeholders for other entities that will exist in various locations and forms in actual implementations.

Generally, Function Y would exist in the Residential Gateway, or possibly partly in the UE, and would accept the input from the upstream network in whatever unicast or multicast form, and present unicast to the downstream clients. It would also handle HTTP proxying and name resolution, as discussed further in Section 6.

Function X would handle the insertion of unicast data into multicast, using some appropriate encapsulation protocol (See Annex A for technical detail on the alternatives). X here is both a multicast router and a content ingestion point. These can be separate functions operated by different organisations. There could be further flexibility in decoupling these. The figures above split X and Y each into two logical elements; these may in fact be single components. Error mitigation, either by packet re-transmission or Forward Error Correction (FEC) is an important feature that will also involve X &Y, plus potentially other components.

6 Signalling and Proxying

A key design principle of the WP5 framework is that client applications should not be required to change their behaviour. When a media player application requests content, it makes a series of HTTP requests, initially for the manifest, and subsequently for the media segments that the manifest refers to. Typical schemes include HLS with MPEG2-TS video, or MPEG-DASH with MPEG-4, however the principle is the same in both cases. The requests are likely to be satisfied by a CDN, by means of HTTP redirects or proxying.

6.1 Message flows

Figure 13 shows the current flow for the manifest-request stage, before actual video is delivered, and with the initial DNS lookups omitted. The principle is that the CSP triggers a redirection to the CDN, then the CDN arranges for its most appropriate node to serve the content. For example, a BBC iPlayer client app could request a manifest from the BBC. This request would be redirected to an Akamai CDN, and further redirection would cause the media segments to be served from a relevant Akamai node (not shown in Figure 13). The CDN may change the redirection dynamically for load balancing or other optimisations.

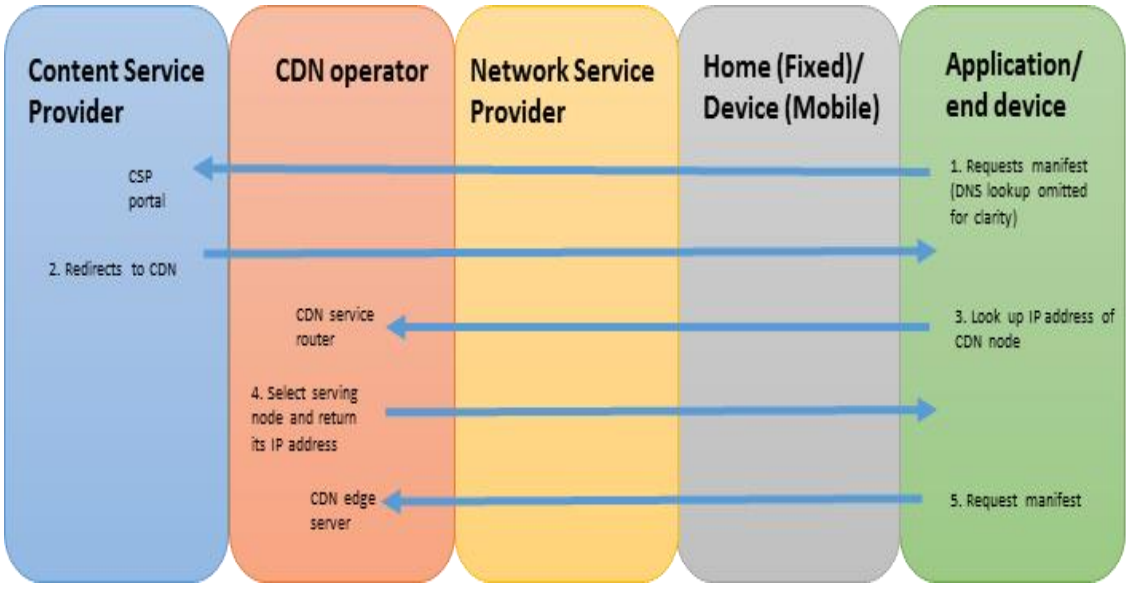

Figure 13 information flow with current CDN

Work package 4 discusses this in more detail, in particular CDN caching and also alternative ways that the physical, transport and application layers of the stack could be developed to become more network-agnostic, particularly regarding the mobile paths. In this document we focus more on the application layer signalling, and Figure 14 shows in simplified form how the paths from Figure 13 could be adapted to support fixed and mobile, multicast or broadcast delivery at the application level. There would of course be significant changes at the lower, network layers, and these are discussed in the WP4 deliverables. The 'user'-level parameters such as content-popularity would steer these network-layer decisions.

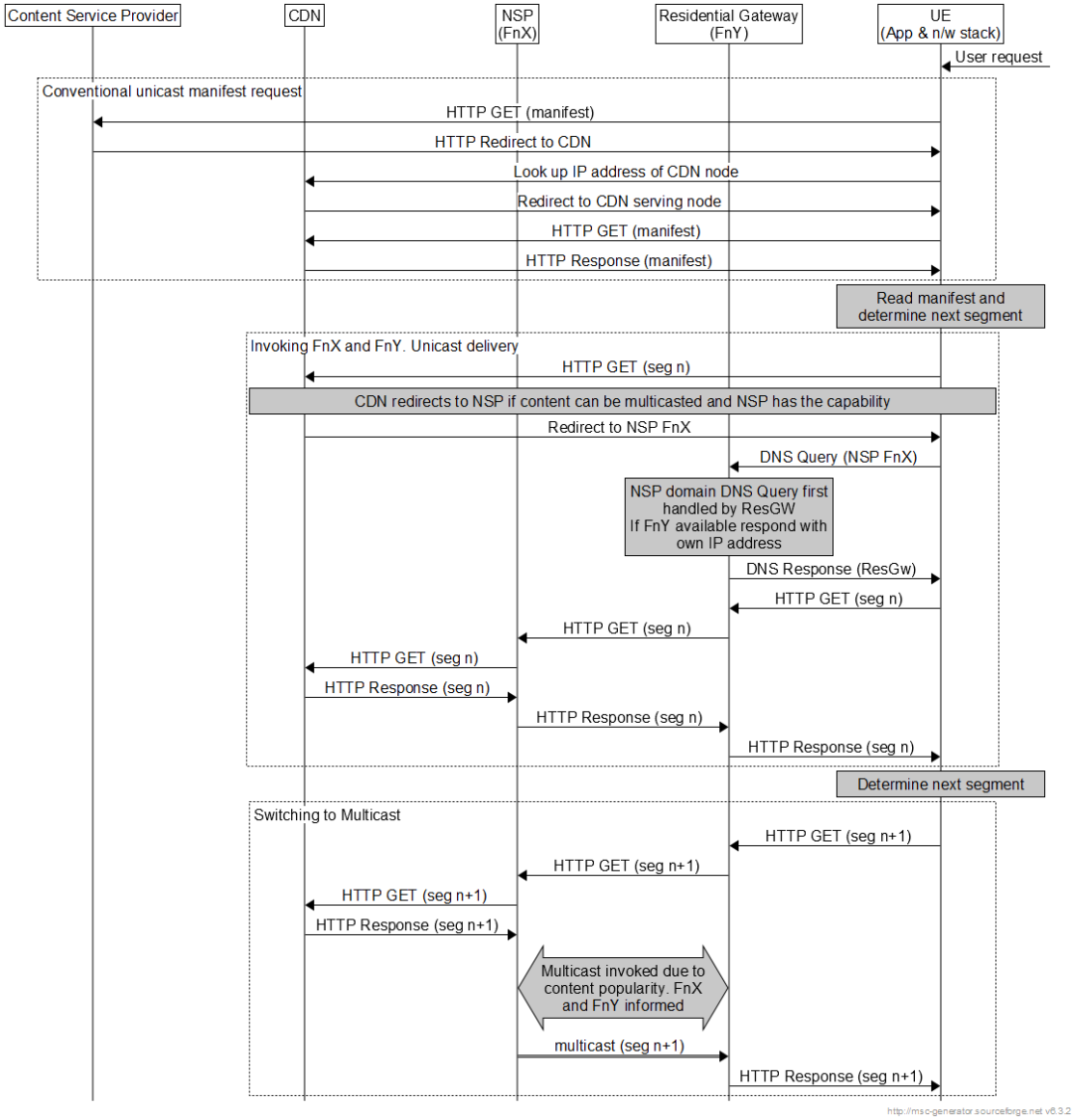

Figure 14 Simplified redirection via the framework

The key change is that a 5G-Xcast logical capability is added to handle the request routing, based on whether content is available on different network types, its popularity, and potentially other factors such as rules. These factors are, from T5.2's viewpoint, external, and could for example be traffic shaping or load management rules specified by a network operator, CDN provider etc. The point here is that the framework supports them, leaving the specification and detail to the implementation work packages.

A challenge with this approach is how to handle the proxying or redirection. Generally, there are two techniques:

- DNS redirection. This is simple and fast, however can't readily cross organisational boundaries, and is therefore probably less suited for a mixed-organisational model.
- HTTP redirection. This is slower since it requires more than one request, however it is able to cross organisational boundaries, allowing a content provider to redirect to their partner CDN for example.

The main research challenge here is how to handle TLS-protected communication. Most current deployments for redirection require TLS certificates belonging to the destination organisation to be deployed on intermediate nodes, which is fine when there is a commercial and technical agreement in place, but impractical without strong trust between these organisations. This could be addressed by for example issuing an HTTP redirect, picking up the certification from the destination redirection.

6.2 Proxy Types

There are a number of possible ways to handle the proxying/redirection. We present alternatives here for information[9], although the engineering details will depend on implementation scenarios.

- In-gateway transparent proxy (this is as the CableLabs spec)
- In-gateway forward proxy
- In-gateway DNS redirection with reverse proxy
- In-gateway DNS interception with reverse proxy
- Explicit HTTP redirection with reverse proxy
- Manipulation of manifest URLs with reverse proxy

There are issues with handling certificates and TLS with some options, and these will need to be investigated in subsequent phases of the project.

[9] List compiled by a DVB working group in TM-IPI as part of the 2017 multicast ABR work; the details are available to DVB members.

7 Dynamic Delivery Mode Selection

The design principles (section 3) indicate that multicast/broadcast should be treated as internal network optimisation. This requires a mechanism to dynamically switch delivery modes from unicast to multicast for popular content. 3GPP Release 14 includes a specification of MBMS Operation On Demand (MOOD) that partly fulfils this role but is limited to use in mobile networks. It is also slow to respond to rising popularity and fairly complex in its architecture. 3GPP MOOD is however worth considering here as a starting point for the wider issue of dynamic switching in the mixed-network contexts being studied as part of 5G-Xcast.

In Annex C we compare 3GPP MOOD with a fixed-network implementation of relevant features, in order to help generate our aspirations for dynamic delivery mode selection in 5G-Xcast. Our current thinking is summarised below:

Property	**5G-Xcast (Aspirations)**
Traffic types	Standards based ABR (DASH, HLS etc.) and generic HTTP/HTTPS, HTTP/2 QUIC
Identify eligible traffic	CP and CDN operator should be able to identify eligible traffic including when carried over HTTPS, and to allow the system to process that identified traffic.
Signalling point-to-multipoint traffic to Function Y	The agent on the Residential Gateway or the middleware of the User Equipment is aware of content available in multicast (via an announcement or marking mechanism). Don't require the end user application to be aware.
Identify audience size (when consuming unicast)	Aim to minimise signalling traffic. Any explicit signalling to be done by Function Y and is transparent to client applications.
Identify audience size (when consuming point-to-multipoint)	Will probably need some form of explicit signalling.
User Equipment location	Is it probably enough to know UE location per access network? For mobile will probably be done via Cell-ID or Service area ID. For fixed it will probably be IP address of client identified by HTTP request.
How Function Y (in UE or Residential Gateway) made aware of switch to unicast	Unicast source should always be available. UE can decide to switch to unicast by abandoning multicast/broadcast. Network can force switch to unicast by removing multicast/broadcast variant. UE forced to consume unicast. Might also want explicit signalling so that buffering can be kept to a minimum?
How Function Y (in UE or Residential Gateway) made aware of availability of point-to-multipoint	Need a mechanism to inform client about multicast/broadcast availability. Could be done by any number of mechanisms e.g. DASH manifest manipulation, HTTP redirect, HTTP header field, explicit signal or at a lower protocol layer (possibly a QUIC mechanism?). Function Y would make the switching decision. Needs further discussion.
Considerations for use with unicast multilink	Need to think about whether the presence of multilink has a bearing. Might need additional interfaces, it might impose limitations etc. Multilink integration point is being discussed in WP4. For now, it is difficult to assess any further.
Implications for object-based content and other non-media content (e.g. software updates)?	Need to consider what entities we are counting when measuring audience size. Entire media stream or individual objects. What is the granularity with which we can switch delivery modes?
Fault handling / management	For further discussion. What are the failure points? How are they handled? What about congestion in broadcast/multicast? Partial coverage etc.

Table 1 Mode Selection Aspirations

Mode selection could be signalled in HTTP by using the *alt-svc* tag or the ALTSVC HTTP/2 Frame[10] to indicate that the content was also available in multicast. This could remove the need for a redirect from the CDN provider to the network operator.

[10] https://tools.ietf.org/html/rfc7838

8 Open Issues

This section presents a list of topics that can't readily be specified in our framework, that implementations will nevertheless have to consider. As 5G-Xcast work progresses, the scope and solutions to some of these should become clearer, and we hope to develop answers to some of them as the project progresses, to be reported in other deliverables. Our recommendations need to be pragmatic in the real world, otherwise commercial organisations will never adopt them. Whilst WP5 is not directly designing systems for billing, service security, legal frameworks, interoperation and standards-compliance, it must be mindful of their existence. The framework must not therefore add breaking changes to these existing functions.

- What knowledge is needed from the media stream to make the unicast/multicast transport decision? e.g. does the network operator need to be able to read manifests, see URLs, receive hints from the content provider etc.?
- Does the manifest need to be modified? E.g. to encourage the client to request only the segment that we have chosen to deliver by MC. We must not modify the actual content en-route.
- Where are the termination points, and what are the trust relationships needed for TLS? With single-ended or mutual authentication?
- At the end device application: how much does the application developer have to know about the network technology? What does the software structure on the end device look like?
- How do we determine what the multicast rate should be without knowledge of the media? Allied to this, the playout rates need to be managed when dealing with a hybrid unicast/multicast/broadcast situation: unicast rates may be highly variable (particularly at start up), and broadcast/multicast rates constant.
- We must deal with session-specific parameters (such as URL signing).
- We should be able to handle both media streaming and also large file distribution.

9 Conclusion

In this document we have presented a set of opinions on the ideal characteristics of a global efficient content delivery architecture. We have justified these opinions by discussing the current technology in use and previous unsuccessful technologies. We have been mindful of both the technical and commercial requirements of the organisations involved in delivering content.

With the existing delivery ecosystems of broadcast, IP-based unicast and multicast and mobile delivery, there are proven technical delivery approaches and commercial models, both within and across the technologies. LTE broadcast/eMBMS is now being adopted, and a set of best-practice recommendations would help integrate eMBMS as an at-scale technology as part of this mix. To be successful, the addition of “yet another” technology should not compel existing business or commercial arrangements to be radically changed: the state of the art for delivery suits the current mix, but a goal for 5G-Xcast is to help ease the insertion of the newer mobile broadcast technologies.

A key premise of 5G-Xcast is for techniques such as multicast to be hidden from the end users (producers & consumers), and to be made available as an internal optimisation; part of the toolkit available to the operators.

Our view is that Content Delivery Networks should continue to be used for global reach, but that multicast or broadcast should be used at the edge where possible. This would be achieved by providing indirect access to the multicast capability, so that it looks logically like an extension to the CDN and is used to optimise the network operator’s network, rather than the need to expose it as a service to be offered in its own right.

This approach has some technical and commercial challenges.

There are several significant technical challenges. Central to the unification concept is that all network types and topologies use the same (or easily convertible) encapsulation formats. The challenge is specifically trying to deliver Internet protocols over broadcast and multicast networks, which traditionally use a very different protocol stack.

Related to this, is that the Internet is really based on unicast protocols, whereas efficient delivery at the edge of the network requires multicast or broadcast. Exploiting a point-to-multipoint distribution capability to make point-to-point flows more efficient is very challenging. It will require delivering streams which are not synchronised in a synchronous manner.

The main commercial challenge is to establish the depth and nature of the relationship with the CDN operator. The business model for integration must be such that both the CDN operator and the network operator benefit.

A Multicast Encapsulation Protocols

In order to transport discrete elements such as files over multicast, some kind of encapsulation mechanism is required. This is because multicast essentially appears as a continuous fixed-rate and unidirectional 'pipe', connecting a single entry point to multiple exit points. If file contents are simply emptied into the pipe, then essential information such as file names, lengths, metadata etc. would be lost. Techniques are therefore required to preserve this information (and to deal with any loss or corruption), such that the original information can be recreated at the destination endpoints of the multicast. With this in place, a unicast end client could for example request unicast, file-based streaming media from a unicast manifest, and have the results delivered in their original unicast format, even though an internal network optimisation had been employed to deliver parts of it over multicast.

This section presents background information on the techniques available. Note that this document does not make specific recommendations for adoption, since this decision depends on the domain (mobile vs. fixed, characteristics of the networks etc.), and is best considered further by the WP3 and 4.

The candidates are:

- RTP Extensions;
- FLUTE;
- ROUTE;
- NORM;
- HTTP/2 with QUIC

Additionally, there is work currently within the DVB on Filecasting and multicast-ABR that we briefly summarise in this section, since there is valuable overlap with the goals of 5G-Xcast.

A.1 RTP Extensions

RTP is generally used to carry payload in multicast. It is a lightweight protocol, and in its basic form used a fixed-size header of 12 bytes, which may be compressed by the network drivers for efficiency. Of interest here is that the header contains a sequence number (allowing lost packets to be identified), and a timestamp (allowing the multicast rate to be expressed). BT's multicast for example comprises a sequence of an RTP header of 12 bytes, followed by seven MPEG2-TS packets of 188 bytes each, all contained in a UDP frame, to give a total of 1328 bytes per packet.

The specification allows for extension headers that may be used to convey additional data. There are three different ways that the header may be extended, and these options permit data such as originating URLs, file names, lengths etc. to be encoded within the header. This is a relatively simple way of carrying encapsulation data and, as such, may be somewhat limited for 5G-Xcast requirements. This simplicity does however make RTP extensions an attractive option for lab-based prototyping, and this may be further explored in Task 5.4 or Work package 6.

A.2 FLUTE

The FLUTE v1 protocol (File Delivery over Unidirectional Transport) protocol is used in eMBMS (RFC 3926[11]) and DVB-H. FLUTE is a massively scalable delivery protocol largely built upon Asynchronous Layered Coding (ALC, RFC 5775) and Layered Coding

[11] https://tools.ietf.org/html/rfc3926

Transport (LCT, RFC 5651); it leverages IP multicast to optimise content distribution. Where ALC provides capability to transport binary objects (files), FLUTE provides the metadata for these binary objects, thereby providing the information for the client on how to receive and process the objects. The transport of this metadata and the actual objects are done by ALC.

ALC combines the LCT (Layered Coding Transport) building block to provide in-band session management functionality, a congestion control building block, and the AL-FEC (Application Layer Forward Error Correction) building block to provide reliability. The AL-FEC building block allows the choice of an appropriate FEC code to be used within ALC, including the possibility of sending the original data without FEC.

For FLUTE, the file is partitioned in one or several source blocks, as depicted in Figure 15. Each source block is split into source symbols of a fixed size. Symbol parameters are signalled in the session setup and are fixed for one session. Then, for each source block, FEC encoding can be applied to generate additional repair symbols. The collection of source and repair symbols is generally referred to as encoding symbols.

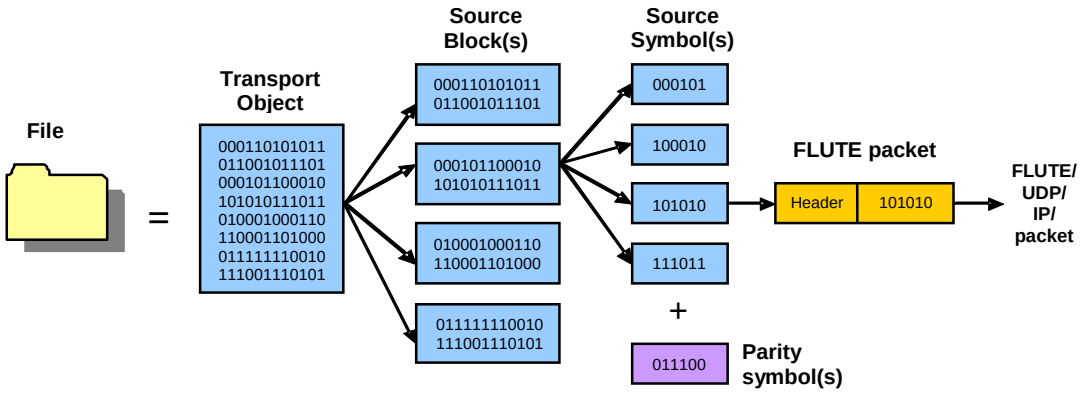

Figure 15 FLUTE blocking algorithm

Each encoding symbol is assigned a unique encoded symbol ID (ESI). If the ESI is smaller than the number of source symbols, then it is a source symbol; otherwise it is a repair symbol. Symbols are either transmitted individually or concatenated and mapped to a FLUTE packet payload. The source block number, the ESI of the first encoded symbol in the packet and other file parameters are signalled in the FLUTE header. FLUTE packets themselves are then encapsulated in UDP and then distributed over IP multicast bearers. Receivers collect correctly received FLUTE packets and with the information available in the packet header and the file delivery session setup, the structure of the source block can be recovered. Application layer FEC can be used to provide FEC protection to the file as an entity. An appropriate combination with link layer FEC can result in very efficient file delivery services.

FLUTE can be used for multicast or unicast file delivery but its primary application is to multicast. Every multicast bearer is implemented through an LCT channel, described by two IP addresses: the first one identifies a multicast group and the second one identifies the sender generating packets towards that particular group. Both Any-Source-Multicast (RFC 1112) and Source-Specific Multicast (RFC 4607) delivery modes are supported. An LCT session can be composed of several LCT channels, having multiple constant or variable transmission rates. Receivers joining a session dynamically select the LCT channels used to fetch packets, according to the perceived congestion state and to the transmission rates of channels. By joining and leaving LCT channels dynamically, each receiver adjusts its reception rate independently of other participants to the session. The

selection of LCT channels is carried out on the basis of a specific congestion control protocol which, in turn, is influenced by the type of delivered content. LCT (RFC 5651) recommends Wave and Equation Based Rate Control (WEBRC – RFC 3738) as the congestion control protocol, even though it may respond slowly to changes in available bandwidth. WEBRC partitions time into time slots and arranges channels into layers on the basis of their speeds, which vary over time slots according to a known pattern (short increase followed by an exponential decrease and a quiescent period [10]. Further congestion control protocols compatible with LCT are reported in [11] and [12]. Regardless of the employed congestion control algorithm, FLUTE is unidirectional as there is no feedback mechanism or channel required from client to sender. The absence of feedback volume plays a key role in FLUTE scalability.

The metadata that describes the file and the delivery of the file are arranged into a File Delivery Table (FDT). Some examples of the metadata are:

- Size of the transmission object and file
- Name, Identification, and Location of file
- Media type of file
- Message digest

Multiple files can be described in a single FDT. The FDT can be sent before the actual transmission is started, repeated during the transmission as well as the end. FLUTE is extensible as the FDT structure described in RFC 6726 can also be enhanced with *new metadata*, something that has been done by 3GPP for eMBMS.

Receivers fully know how to interpret transmission objects after receiving an FDT. Without an FDT those transmission objects could be cached until the FDT is received.

FLUTE also allows files to be repeated carousel-fashion and updated (by signalling new versions) as well as expiry of files.

A.3 ROUTE

Real-time Object delivery over Unidirectional Transport (ROUTE) is used as the transport for ATSC 3.0 systems.

To some extent, ROUTE may be considered backwards-compatible with FLUTE. However, ROUTE uses FLUTE v2 or ALC / LCT version 2, whereas 3GPP FLUTE uses v1 of these, and this means that ROUTE could not be directly used to interoperate with 3GPP-FLUTE systems.

Originating from FLUTE, ROUTE also is an object-based broadcast delivery protocol, which employs LCT version 2 and ALC v2 for defining the source protocol, and AL-FEC for defining the repair protocol. The basic information on these are as above mentioned in Section A.2. In this section, fundamental differences are explained for better understanding of the improvements.

A goal in developing ROUTE was to improve on FLUTE regarding timeliness of media delivery, by offering:

- Real-time delivery of object-based media data,
- Flexible packetisation, including enabling media-aware packetisation as well as transport-aware packetisation of delivery objects
- Independence of files and delivery objects, i.e. a delivery object may be a part of a file or may be a group of files.

Figure 16 illustrates a basic functional difference between the two protocols regarding the FDT functionality. ROUTE extends the FLUTE FDT by providing optional additional

rules and metadata that can be used by the ROUTE receiver along with the LCT header to generate location information of the objects on-the-fly. In this way, the crucial need for continuous sending of the FDT information can be avoided. This extended functionality is called Extended FDT (EFDT).

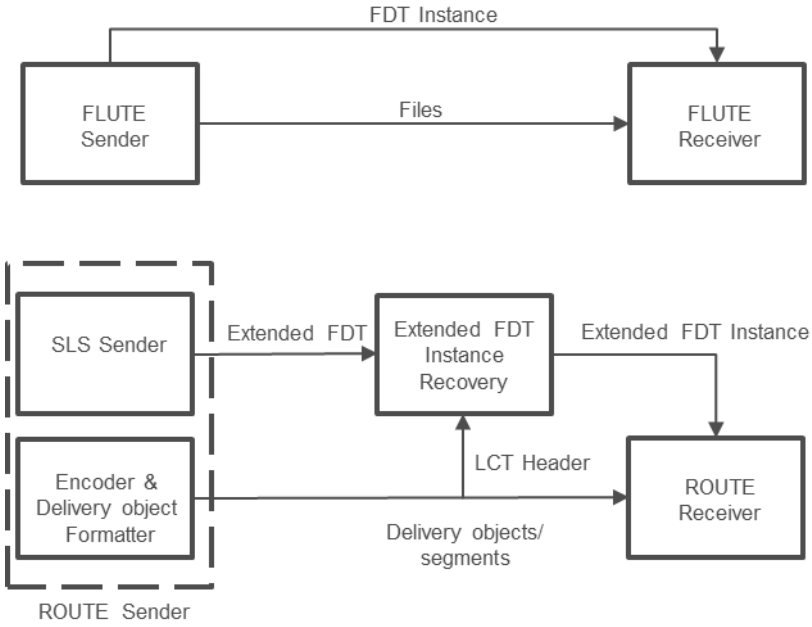

Figure 16 ROUTE distribution in file mode in comparison to FLUTE distribution[12]

ROUTE is split into two main components:

The source protocol for delivery of objects or flows/collection of objects, through which it defines not only the sender and receiver operation but also the object packaging. For this purpose, ROUTE employs LCT and ALC protocols. Use of well-established protocols eases the process of extension for future improvements.

- The repair protocol for flexibly protecting delivery objects or bundles of delivery objects that are delivered through the source protocol. It is AL-FEC based and makes use of IETF FECFRAME (RFC 6363) framework with the distinction that ROUTE protects delivery objects rather than packets. Hence, it enables better time diversity and effectiveness [7]. The source protocol is independent of the repair protocol, i.e. the source protocol may be deployed without the ROUTE repair protocol. Repair may be utilised only for certain constrained or specialised deployment scenarios, for example only for mobile reception, only in certain geographical areas, only for certain service, etc.

Several papers have been published on ROUTE/DASH efficiency, e.g. [13].

A.4 NORM

The NACK-Oriented Reliable Multicast Transport Protocol (NORM, RFC 5740) provides efficient and scalable delivery for bulk data and streams. NORM supports reliable data delivery over IP multicast but also supports unicast (point-to-point) data transfers; in particular, the protocol is compatible both with Any-Source-Multicast (RFC 1112) and Source-Specific Multicast (RFC 4607) delivery modes. NORM receivers joining and

[12] Adapted from ATSC Standard: "Signalling, Delivery, Synchronization, and Error Protection," A/331:2017, 6 December 2017.

leaving multicast groups can be subject to application policies, aiming at limiting the risks of multicast performance disruption (possibly by excluding badly behaving receivers).

NORM operates on top of the User Datagram Protocol (UDP) and supports reliability via a NACK-based Automated Repeat Request (ARQ) that uses packet erasure coding for very efficient group communication: in addition to the Multicast NACK Building Block (RFC 5401), NORM implements the building block recommendations of Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfers (RFC 3048), including a congestion control mechanism based on a source-based delivery rate regulation (TCP Friendly Multicast Congestion Control – TFMCC RFC 4654) and making use of FEC codes (FEC Building Block, RFC 5052) to proactively react to information loss.

NORM provides for three types of bulk data content objects (NormObjects) to be reliably transported. These types include:

1. Static computer memory data content (NORM_OBJECT_DATA type);
2. Computer storage files (NORM_OBJECT_FILE type);
3. Non-finite streams of continuous data content (NORM_OBJECT_STREAM type).

NORM does not explicitly provide for global or application-level identification of data content (“metadata”) within its message structures. However, the NORM_INFO message can be leveraged by the application for this purpose if desired, or identification can alternatively be embedded within the data content.

TCP-friendly congestion

NORM provides for automated TCP-friendly congestion control and mechanisms to support end-to-end flow control; this is realised by identifying a particular steady-state sender transmission rate that fairly competes with TCP flows (RFC 4654, [14]). In addition to its TCP-friendly congestion control, NORM can also be configured for fixed-rate operation and the U.S. Naval Research Laboratory implementation supports some additional automated congestion control options suitable for use in bit error prone wireless communication environments. The regulation of a sender delivery rate is based on the analysis of two factors:

1. the estimation of packet loss information in the population of receivers and
2. the evaluation of Round-Trip Times (RTTs) of bottleneck paths within the multicast topology.

The aim is to identify the maximum transmission rate that can be sustained by the participants to the session, whose supremum is constrained by the receiver with the lowest transmission rate (current limiting receiver - CLR). Such a process is carried out by stimulating explicit feedback from the population of receivers through a periodic injection of specific messages from the source, which initiate the measurement of RTTs.

NACK-based mechanism

NORM reliable ARQ operation is principally NACK-based (negative acknowledgement when packet loss is detected). Receivers start a NACKing procedure to request a repair operation; this activity is controlled by a backoff timer (RFC 5401), which postpones the transmission of a NACK message by taking into account the size of the receiver population and the maximum RTT between the sender and receivers. In such an interval each receiver accumulates multiple pending repair requests that are finally sent as aggregated messages, reducing the risk of NACK implosion for the sender. By resorting to a similar mechanism, the sender postpones the beginning of a repair operation in order to accumulate and aggregate several repair requests, determining an optimal

repair strategy for the whole population of receivers. At the end of this period, the sender "rewinds" the state of transmission by transmitting the necessary repair messages.

NORM also supports optional positive acknowledgment (ACK) from receivers that can be used for delivery confirmation and explicit flow control. The positive acknowledgment procedure is essential in the aforementioned congestion control mechanism, as it is used to carry on explicit feedback to assess RTTs within the population of receivers. An additional type of acknowledgment can be generated to explicitly notify the completion of reliable data reception. Finally, further acknowledgment types can be defined by applications for ad hoc purpose.

Issue with NACK protocols

NACK-based protocols are more likely to be used in multicast transmission, since they don't generate a large volume of acknowledgement packet traffic when reception conditions are good. In the case where there are many receivers, a constant stream of positive ACKs could impair the sender's performance. However, there still exists the potential problem of so-called "NACK implosion" when suddenly a huge volume of retransmission requests overloads the sender.

To overcome this risk, the Naval Research Laboratory's NORM implementation can instead be configured to provide a basic UDP-like best effort transport service (with no receiver feedback) and this can be enhanced by adding an application-configurable level of proactive Forward Error Correction (FEC) packets to the transmission. By default, NORM only sends reactive FEC repair packets in response to NACKs, but can also be configured to proactively send additional redundant repair packets for a level of reliability without any feedback from the receivers. The cost of this redundancy is a bit rate overhead in the NORM transmission.

Applicability to multicast ABR

The limitation with NORM in its basic definition is its scalability. With the Naval Research Laboratory implementation configured to use proactive FEC, NORM is better adapted to multicast ABR use.

However, the TCP-friendly congestion mechanism is not applicable in this context. In ABR streaming, the receiver instead adapts itself to the available networks bandwidth by selecting a multicast flow with a different bit rate.

A.5 HTTP/2 and QUIC

A.5.1 HTTP/2

HTTP/2 was designed to address limitations of previous HTTP versions (HTTP/1.0 and HTTP/1.1) and to improve application performance. The biggest drawback of HTTP/1.1 is its limited support of concurrency between the requests. HTTP/1.1 introduces request pipelining but the response may be subject to head-of-line (front-of-queue) blocking. Requests must be responded to in the same order as they are received and head-of-line blocking occurs when a server needs more time to respond to earlier received requests. Another issue with previous versions of HTTP is verbosity because the protocol is text-based rather than binary-encoded. Moreover, HTTP header fields are often repeats. This causes unnecessary traffic which may lead to poor TCP performance due to TCP congestion algorithm implementation.

HTTP/2 addresses these issues by:

- Interleaving (multiplexing) request and response messages on the same connection

- New efficient binary encoding of all protocol messages
- Request prioritisation
- Binary compression of headers.

It is worth noting that HTTP/2 does not change the semantics of HTTP. HTTP/2 addresses the inefficiencies in the underlying transport of application request and response messages with the outcome of fewer separate TCP connections being required to achieve the desired performance and hence better network resource utilisation.

HTTP/2 introduces the concept of streams into the protocol. Each HTTP request and response exchange is associated with own stream. The basic protocol unit is a frame which are of several types (DATA, HEADERS, PRIORITY, RST_STREAM, SETTINGS, PUSH_PROMISE, PING, GOAWAY, WINDOW_UPDATE and CONTINUATION).

The multiplexing within the connection happens at the stream level. Endpoints are free to interleave frames from multiple concurrent streams in a single HTTP/2 connection. The stream identified by ID 0x0 is a special stream used only for frames associated with the connection. The order of frames is significant in a stream, i.e. the endpoint must closely follow the specified order of the frames.

The multiplexing of steams over one HTTP/2 (TCP) connection introduces contention. HTTP/2 therefore introduces application-level flow control to address the contention. The flow control applies both to the individual streams and to the HTTP/2 connection as a whole. The flow control is managed based on window size communicated using the WINDOW_UPDATE frames. The flow control is also influenced by stream priorities and stream dependencies.

The HTTP request and response exchange over HTTP2 is initiated by a client, which sends an HTTP request on a new stream using a previously unused stream ID. The server responds with an HTTP response on the same stream ID.

Finally, HTTP/2 introduces a push mechanism which allows a server to pre-emptively send additional responses to a client in connection with an initial client request. This mechanism is useful in situations where a server knows, based on the client-initiated request, that additional dependent resources will be needed by the client. This avoids the need for the client to explicitly request the additional resources, thereby saving half a network round trip. The push mechanism could improve application performance in cases of very limited uplink from a client. It is possible for a client to request a server to disable this feature. The push mechanism uses the PUSH_PROMISE frames.

A.5.2 QUIC

Quick UDP Internet connections (QUIC) is a new transport protocol currently under development in the Internet Engineering Task Force (IETF). This section describes the status of IETF standardisation of QUIC.

The QUIC transport protocol addresses the latency of connection establishment of TCP/TLS. It provides data multiplexing based on a streams concept similar to that of HTTP/2. Stream multiplexing in the transport protocol can tackle the head-of-line blocking issues that may arise with earlier HTTP/1.1 and HTTP/2. QUIC uses UDP as the underlying transport protocol, and can therefore be implemented outside the operating system kernel, in user-mode code. Having the network stack implemented in the kernel means that it must be general-purpose, and development and evolution is slow, since absolute reliability of the whole system is a major priority. In contrast, allowing the QUIC protocol to reside inside the application (the only kernel requirement is to implement the lightweight and simple UDP elements) means that its evolution is in the scope of application developers, and may be much more rapid.

Although UDP traffic may be blocked in the Internet, indications are that the amount of blocked UDP traffic is relatively low, at between three to five percent[13]. Higher layers utilising QUIC must be able to switch to a fallback transport solution, e.g. TCP, in the case that a QUIC connection cannot be established.

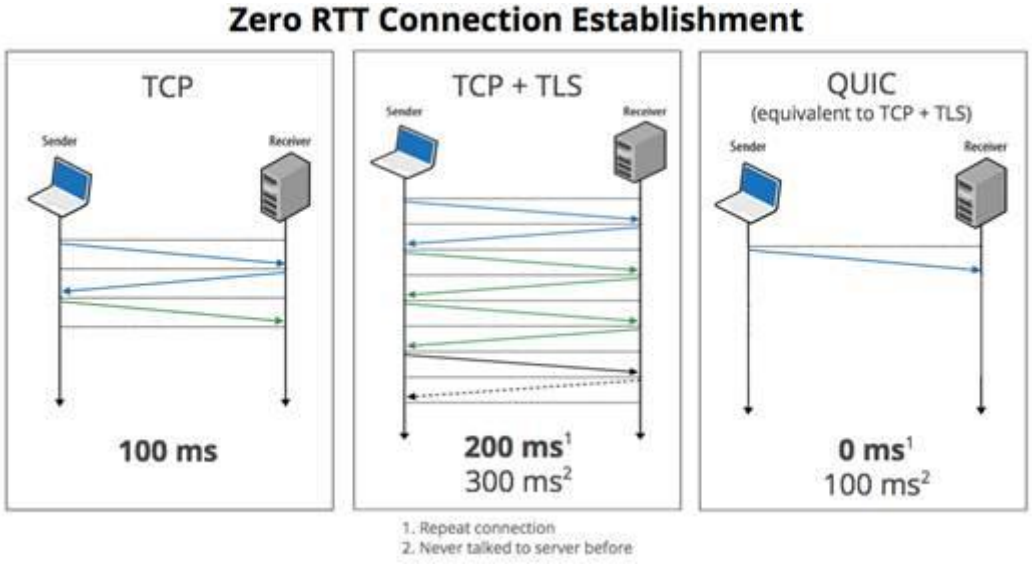

Figure 17 QUIC fast connection setup time

QUIC defines the UDP payload to be QUIC packets which have a well-defined structure using either short or long header format. The short format is used during the initial connection handshake; thereafter the short header format is used. Apart from the initial handshake, all QUIC packets are encrypted. There are several packets types defined for the various needs of the protocol. The payload of packets after decryption of protected payloads are QUIC frames. The current draft specification defines 15 frame types, of which 14 types are used for control purposes. User data are carried by the STREAM frames.

Although QUIC is a transport protocol and HTTP/2 is an application protocol, QUIC streams borrow many concepts from HTTP/2 streams. Each QUIC stream is a lightweight, ordered byte-stream which may be bi-directional or unidirectional. Each stream is identified by a unique ID, used to label QUIC packets travelling in either direction, which cannot be reused during a particular QUIC connection due to cryptographic constraints. There is a dedicated stream for the cryptographic handshake at the start of a connection. Each stream is a subject to individual flow control. The level of concurrency within the QUIC connection is limited by the maximum number of stream identifiers used at once. QUIC does not define frames for exchanging priority information like HTTP/2 does. Instead it relies on the receiving priority information from the application that uses QUIC. For example, if the client application is HTTP then the HTTP endpoints may exchange the priority information using the PRIORITY frames, see below. The flow control of QUIC connection is then the combination of steam flow control and stream prioritisation.

QUIC provides a general transport over UDP. The HTTP over QUIC draft defines the mapping of HTTP semantics over QUIC[14]. The availability of QUIC transport may be advertised to a client in either alt-svc header of an HTTP/1.1 response or in an HTTP/2

[13] https://tools.ietf.org/html/draft-ietf-quic-applicability-00

[14] https://tools.ietf.org/html/draft-ietf-quic-http-07

ALTSVC frame. This HTTP client is then free to establish an HTTP/QUIC connection in parallel to the existing legacy HTTP connection, or instead of it. As mentioned above, the QUIC protocol may experience problems with the connection establishment due to blocking by middle boxes. In this case, the client should continue to use the existing connection or try another endpoint offered by the origin.

In short, the HTTP mapping over QUIC reserves stream with ID 0x1 for HTTP SETTINGS and PRIORITY frames. The HTTP message exchange comprises an HTTP request transmission from a client on a new QUIC stream ID, followed by an HTTP response transmission from a server on the same stream ID. A new binary framing layer for HTTP is needed due to the fact that the QUIC transport layer subsumes certain concepts that were previously provided by HTTP/2. The example reasons for the new framing are a) HTTP frames are already on a stream and thus they can omit stream number; b) END_STREAM flag is not required because stream termination is handled by QUIC. Many differences in framing arise because HTTP/2 provides an absolute ordering between frames across all streams, while QUIC provides this guarantee on each stream only.

HTTP over multicast QUIC

Another IETF draft specification introduces the possibility to transport HTTP over IP multicast using QUIC [15]. The concept of a QUIC connection does not suit unidirectional transfer, and so the concept of a multicast QUIC session is introduced instead. The session is fully-established when the session has at least one sender and at least one receiver.

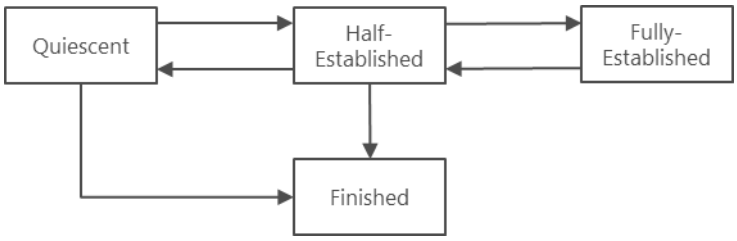

Figure 18 Multicast QUIC session states

The availability of a multicast QUIC session may be advertised to potential receivers in Alt-Svc HTTP headers, as shown in step 2 in Figure 19. The advertisement is sent in the response to the client's unicast HTTP request, step 1, meaning the client must first communicate with the server over unicast. In order to provide additional security safeguards, it is recommended to advertise multicast QUIC session availability over a secure transport, such as HTTPS. An HTTP response may include multiple values in the Alt-Svc header to advertise multiple multicast QUIC sources. The server's preference is indicated by the order of values with first value being the most preferred alternative [16]. The server may indicate its preference for multicast QUIC and a gracious client implementation should follow the server's preferences. Upon the receipt of the advertisement, the client can join the IP multicast group (e.g. IGMPv3 or MLDPv2), step 3, and then receive HTTP content from the multicast QUIC session sent as HTTP/QUIC PUSH_PROMISE frames.

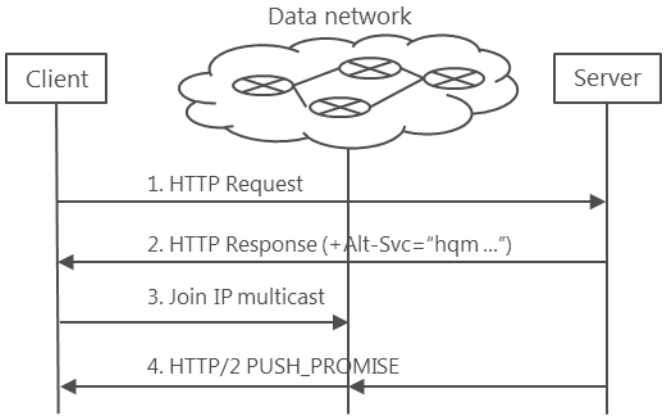

Figure 19 Multicast QUIC session advertisement and HTTP content transfer

The loss recovery mechanisms include currently only unicast repair using HTTP range request. The IETF may standardise FEC schemes for QUIC in future, at which point they could also be used by multicast QUIC

A.6 (DVB) Filecasting

The DVB project is evaluating use cases for distributing file-based media over broadcast, termed Filecasting. Since this is internal DVB work, private to the DVB, we simply present a short summary here.

Filecasting requires similar techniques to the encapsulation methods discussed earlier; many of the same protocols (FLUTE etc.) could be candidates, and will need to be evaluated. Similarly, the interface between the CDN and Filecasting links will need to be studied, (S)FTP, HTTP(S), multicast etc. being candidates for specific transports. The DVB project is also currently working on gap analysis. These three areas are of course key to 5G-Xcast in general, and WP5 in particular. Some 5G-Xcast partners are also DVB members, and we will continue to monitor the state of the DVB Filecasting work.

A.7 (DVB) Multicast ABR

The DVB Technical Module TM-IPI has been working on a reference architecture for multicast ABR (m-ABR), and in February 2018 has published a "Blue Book" recommendation defining a logical reference architecture[15]. Once again, there is overlap in methodology and technology, particularly around the encapsulation protocols and gap analysis. The DVB work continues, and as of September 2018 its focus is in two areas: evaluating the encapsulation protocols for the multicast-delivered content, and on standardising the control-plane interfaces between functions X and Y. Separately, the Broadband Forum (BBF) are also studying m-ABR and at time of writing, the DVB and BBF have a formal liaison agreement for joint study. The BBF have adopted YANG (Yet Another Next Generation) for data modelling in their study of the m-ABR control interfaces. The data and control interfaces are key parts of these hybrid multicast/unicast/broadcast systems, and these initiatives are being tracked as part of 5G-Xcast.

[15] https://www.dvb.org/news/dvb-releases-reference-architecture-for-ip-multicast

B Multilink

In this section, we use the term multilink (ML) to refer to various combinations of IP links aggregation. For example:

1. 5G cellular network link of one operator with a Wi-Fi network link of the same or another operator.
2. 5G cellular network link of one operator with a cable, xDSL or satellite IP link of same or another operator.
3. Any number of IP links of same or different operators

Multilink comprises a set of technologies that allow content to be split over multiple delivery paths, and recombined before delivering to a client.

Multi-link aggregation is currently implemented only for unicast streams, bringing together distinct unicast connections to support a stream. One of the challenges in 5G-Xcast is to see how a sophisticated non-naive broadcast or multicast strategy over multi-link can be achieved. In this case, 5G-Xcast will benefit not only from seamless transitions between broadcast and multicast to unicast (and vice versa), and/or seamless unicast experience side by side with broadcasts to others, but also higher-quality broadcast will be enabled by using multiple connections. The other challenge of 5G-Xcast is related with the specific usage of multi-connectivity in wireless.

Deliverable D4.1 explains Multilink in more detail, and discusses how it could be incorporated into 5G-Xcast.

B.1 Multipath TCP

User plane aggregation among multiple novel 5G radio technologies could take place on PDCP level (or their 5G equivalents), IP layer or transport layer, for example MPTCP (Figure 20).

Multipath TCP (MPTCP) is an ongoing effort of the Internet Engineering Task Force's (IETF) Multipath TCP working group that aims at allowing a Transmission Control Protocol (TCP) connection to use multiple paths to maximise resource usage and increase redundancy.

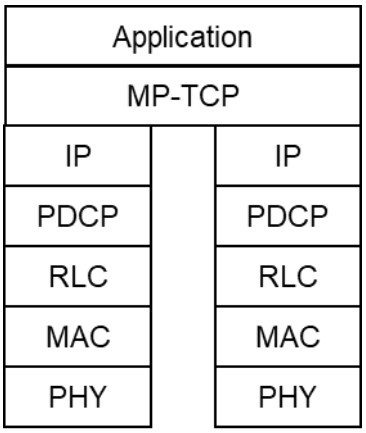

Figure 20 Multipath TCP protocol stack

Figure 21 shows the logical architecture of this solution.

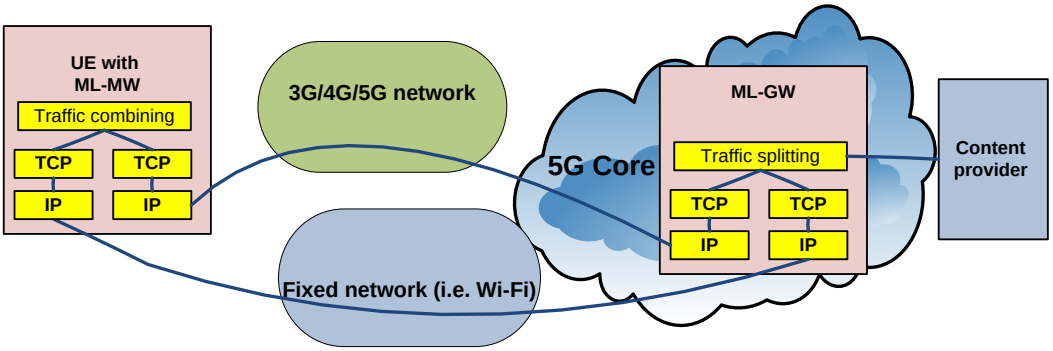

Figure 21 MPTCP logical architecture

The connectivity between the ML-MW and the ML-GW is established using a Layer 4 multipath transport service enabling IP flows to use multiple paths in the Hybrid Access path group simultaneously. As an example, a L4 multipath implementation using MPTCP sets up multiple TCP sub-flows over the different access networks and utilises real time ML-MW to ML-GW flow control. The ML-MW and ML-GW are responsible for managing the MPTCP paths, including establishment and tear down. The implementation itself is access network agnostic, therefore no changes at either the fixed broadband or the 3GPP access networks are necessary. The ML-MW and ML-GW terminate the end user layer 4 sessions before transporting the data over the Hybrid Access paths, effectively executing a proxy function for the end user sessions.

C Dynamic Delivery Feature Identification

Table 2 summarises relevant features from 3GPP MOOD, and also how, as an aspiration, they may be applied in the context of the mixed fixed/mobile context. We have used information from a proprietary implementation as examples of how some of the 3GPP features may be realised on the fixed network, since this gives us a good context to generalise from the 3GPP and fixed situations in order to arrive at the mixed case.

This is used to frame the more general discussion in Section 7.

Table 3 lists some of the important factors that multicast encapsulation protocols must address, and how these are handled in the various alternatives of FLUTE, ROUTE, NORM and HTTP/2-QUIC. As already stated, further analysis will be required to make an informed decision for deployment, and this will be addressed in the other technical work packages.

Property	3GPP MooD	Proprietary Solution	5G-Xcast Converged Aspirations
Traffic types	HTTP/HTTPS(MBMS download delivery) generally DASH RTP/RTSP (MBMS streamed delivery)	Standards based ABR (e.g. DASH, HLS etc) and generic HTTP/HTTPS	Standards based ABR (DASH, HLS etc) and generic HTTP/HTTPS, HTTP/2 QUIC
Identify eligible traffic	MooD header field in HTTP header	Based on configuration by the operator.	CP and CDN operator should be able to mark eligible traffic including HTTPS
Signalling multicast/broadcast traffic to function Y	Eligible content may be signalled to the client in MooD Configuration Management Object (MO) via OMA-DM (Device Management) Or listed within the service announcement	The end-user never knows Option 1: The agent on the CPE is aware for contents available in multicast – service announcement mechanism Option 2: The end-users request the service router to get contents available in multicast	The agent on the CPE or the middleware of the device is aware for contents available in multicast – service announcement mechanism. Don't want end user application to be aware.
Identify audience size (when consuming unicast)	In either UE-Elected or Network-Elected, the <LocationType> MooD header field is important for UE location and audience size measurement. Fields in Consumption Report messages can also be used for audience size measurement	Service router counts requests No explicit feedback from clients to inform switching decisions	Aim to minimise signalling traffic Any explicit signalling to be done by function Y and is transparent to client applications
Identify audience size (when consuming multicast/broadcast)	Consumption Reports explicitly sent periodically by clients to consumption report server	Agent (function Y) sends Info to service router	Will probably need some form of explicit signalling
UE location	The network may obtain UE location (cell ID or SAI) from - UE per operator's policy - MooD header - LCS procedure in TS 23.271 - consumption report message	IP address of client identified in HTTP request	Is it probably enough to know UE location per access network. For mobile will probably be done via Cell-ID or Service area ID. For fixed it will probably be IP address of client identified by HTTP request
How function Y (client or HGW) made aware of switch to unicast	Several possibilities - The function Y detects that TMGI for a given service is not available anymore - The function Y detects that the UE moves out of broadcast zone - The network announces the stop of broadcast session by updating the session metadata	Data is no longer sent on multicast. Agent detects no data arriving and makes unicast request. Buffer in agent ensures seamless switch.	Unicast source should always be available. UE can decide to switch to unicast by abandoning multicast/broadcast. Network can force switch to unicast by removing multicast/broadcast variant. UE forced to consume unicast. Might also want explicit signalling so that buffering can be kept to a minimum?
How function Y (client or HGW) made aware of switch to multicast/broadcast	UE Elected: Unicast requests sent to network proxy server. MooD redirect response (in headers) activate client MBMS receiver. UE decides when it is happy to switch	Dynamic switch planned in H2 2018. All unicast traffic has to go through to the HGW	Need a mechanism to inform client about multicast/broadcast availability. Could be done by any number of mechanisms e.g. DASH manifest manipulation, HTTP redirect, HTTP header field, explicit signal or at a lower protocol layer (possibly a QUIC mechanism?). Needs further discussion.

Property	3GPP MooD	Proprietary Solution	5G-Xcast Converged Aspirations
Considerations for use with unicast multilink	n/a	n/a	Need to think about whether the presence of multilink has a bearing. Might need additional interfaces, it might impose limitations etc. Multilink integration point is being discussed in WP4. For now it is difficult to assess any further.
Implications for object-based content and other non-media content (e.g. software updates, PWS)?			Need to consider what entities we are counting when measuring audience size. Entire media stream or individual objects. What is the granularity with which we can switch delivery modes?
Fault handling / management			For further discussion. What are the failure points? How are they handled? What about congestion in BC/MC? Partial coverage etc.

Table 2 Dynamic delivery features in 3GPP, fixed and mixed networks

	FLUTE	ROUTE	NORM	HTTP/2 + QUIC
Spec	RFC 6726. Flute is specified for MBMS.	No RFC. Adopted by ATSC 3.0 (along with MMT). ROUTE is based on 3GPP MBMS Download Delivery protocol. Parts are common to FLUTE. Also 3GPP TS 26.346 and principles from FCAST	RFC 5740, adopted for Cablelabs.	Draft IETF, January 2016, also QUIC Crypto and QUIC Loss Recovery drafts.
Relies on	ALC, which relies on LCT, FEC BB, CC BB and Auth BB	ALC, which relies on LCT, FEC	FEC, RTT Collection, Group size est.	Chromium code base, FEC, HTTP/2
Scalability	Unlimited. Massively scalable	Unlimited. Massively scalable	Limited to 10's of thousands. Throughput reduces with 1/SQRT(num_rx)	Intended for unicast but appears usable over multicast, and therefore scalable.
Bi-directional	No – No feedback from receivers possible	No	Yes – receiver can NACK in unicast or multicast back channel. Source can also request explicit ACKs.	Yes – ACK framing more efficient than TCP's SACK. HTTP over multicast QUIC unidirectional (ACKs are prohibited).
FEC	Optional but necessary for (partially) reliable service. Various / configurable. For eMBMS Raptor is mandated	Optional. Used to protect 'delivery objects' rather than packets. Has concept of 'source' and 'repair' objects. Uses AL-FEC such as RaptorQ	Optional. Various /configurable. Default scheme allows sender to adapt block size so that more parity can be generated in response to high loss rates.	Simple XOR FEC protects packet group.
Congestion Control	Optional. Client driven multi-layer approaches that are feedback-free. No CC specified for use in 3G MBMS and DVB-H where channels are pre-allocated. In case of congestion control, receivers must implement it to join a session.	Optional. Client driven multi-layer approaches that are feedback-free.	Optional. 2 specified. Sender rate determined by collective feedback from clients. Rate ultimately determined by slowest receiver.	"Equivalent to TCP": a re-implementation of TCP Cubic, although Google are investigating alternatives. Streams and Connections have separate flow control.
Reliability	Reliant on FEC and repetition in carousel scenarios. Additional out-of-band repair mechanism defined in MBMS	'Repair' protocol is optional. Repair uses FEC	FEC and re-transmission based on NACKs	FEC and re-transmission based on NACKs In HTTP over multicast QUICK data may be recovered by transmitting conventional unicast HTTP requests to the origin server.
Authentication	Optional TELSA. Packet source authentication and integrity check possible.	As FLUTE.	Optional TELSA. Packet source authentication and integrity check possible.	Address-spoofing protection, certificates etc. TLS-protected payloads are always encrypted.
File meta-data	URI, file size, content type, content encoding and MD5. File Delivery Table (XML) broken into instances and transmitted in ALC object 0.	As FLUTE but more metadata available to better-handle different formats, channel change etc.	Can be carried in-band in designated INFO packets. Format is application specific. Several message types: NORM_INFO may be used to carry MIME type info, allowing clients to decide whether to process the	HTTP/2 headers.

			actual data for example. Data typically carried in NORM_DATA messages, with further sub-message types.	
Session/ multicast advertisement	Needs an additional advertisement protocol (e.g. SAP - RFC 2974, HTTP, etc.).		Needs an additional advertisement protocol (e.g. SAP - RFC 2974, HTTP, etc.). Could use HTTP/2 PushPromise to signal URLs. (Cablelabs ChannelMap may be too inflexible[16]).	In HTTP over multicast QUICK, Alt-Svc (RFC 7838) HTTP headers can be used to advertise multicast service from a unicast service.
General	Intended as a file delivery protocol, not explicitly for real-time video, although this should be feasible. Use cases tend to feature carousel-based repeats of file transmission. Due to the lack of feedback, the transmission rate is not self-adaptable and the congestion control algorithm may respond slowly to changes in the available bandwidth. With variable transmission rates, a rigorous planning of channel rates is required to assess the global resulting transmission rate.	Media-aware content delivery facilitates fast channel change.	More complex than FLUTE. Receivers need to have comparable reception rates. Not massively scalable The congestion control algorithm automatically converges to the maximum transmission rate supported by the receiver associated with the lowest bandwidth.	Appears very promising, however QUIC still evolving. Secure, efficient, low start-up time, scalable. Should be operable over multicast. Connection not defined by IP:port, so may be possible to handover without breaking.
Available for evaluation?	One open-source GPL, unchanged since 2007	One, work in progress.	One open-source, no restriction	One, BSD, as part of Chrome Development

Table 3 Encapsulation protocols summary

[16] Discussion with Thomas Swindells (Nokia)

10 References

[1] N. Nouvel, Ed., “Content Delivery Vision,” Deliverable D5.1, 5G-PPP 5G-Xcast project, Nov. 2017.

[2] D. Ratkaj and A. Murphy, Eds., “Definition of Use Cases, Requirements and KPIs,” Deliverable D2.1, 5G-PPP 5G-Xcast project, October 2017.

[3] D. Gomez-Barquero, D. Navratil, S. Appleby and M. Stagg, “Point-to-Multipoint Communication Enablers for the Fifth-Generation of Wireless Systems,” IEEE Communications Standards Magazine, vol. 2, no. 1, pp. 53-59, March 2018.

[4] G. Bumgardner, “RFC 7450, “Automatic Multicast Tunneling”, February 2015”.

[5] B. Kovacs, ““Future Is Near For Object-Based Broadcasting” (2016) [Online]. Retrieved from: https://www.tvtechnology.com/broadcast-engineering/future-is-near-for-objectbased-broadcasting”.

[6] “ATSC. (2017). “ATSC Standard: Signaling, Delivery, Synchronization, and Error Protection”. Retrieved from: https://www.atsc.org/wp-content/uploads/2017/12/A331-2017-Signaling-Deivery-Sync-FEC-3.pdf”.

[7] G. Walker, T. Stockhammer, G. Mandyam, Y. Wang and C. Lo, “ROUTE/DASH IP streaming-based system for delivery of broadcast, broadband, and hybrid services,” IEEE Transactions on Broadcasting, March 2016.

[8] T. Tran, Ed., “Mobile Core Network,” Deliverable 4.1, 5G-PPP 5G-Xcast project, June 2018.

[9] J. Hart, Ed., “Converged Core Network," Deliverable 4.2, 5G-PPP 5G-Xcast project, Oct. 2018.

[10] V. Goyal, ""On WEBRC Wave Design and Server Implementation," Digital Fountain Technical Report no. DF2002-09-001, available at http://www.digitalfountain.com/technology/

[11] L. Vicisano, L. Rizzo and J. Crowcroft, “TCP-like Congestion Control for Layered Multicast Data Transfer," Proc. IEEE Infocom, San Francisco, CA, March 1998.

[12] J. Byers, M. Frumin, G. Horn, M. Luby, M. Mitzenmacher, A. Rotter and W. Shaver, “FLID-DL: Congestion Control for Layered Multicast," Proc. Second International Workshop on Networked Group Communications (NGC), Palo Alto, CA, 2000.

[13] N. Bhat H M and W. Zia, “Optimization of Tune-in and End-to-end Delay in DASH Broadcast over ROUTE,” Proc. IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Valencia, Spain, 2018.

[14] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, ""Modeling TCP Throughput: A Simple Model and its Empirical Validation," Proc. ACM SIGCOMM, 1998.

[15] R. Bradbury and L. Pardue, “Internet-Draft, Hypertext Transfer Protocol (HTTP) over multicast QUIC," draft-pardue-quic-http-mcast-01, August 11, 2017.

[16] M. Nottingham, P. McManus and J. Reschke, RFC7838, “HTTP Alternative Services,” April 2016.