CoDaaS: Content Delivery as a Service for User Generated Contents

Yonggang Wen, Peng Sun, Jianfei Cai
School of Computer Engineering, Nanyang Technological University, Singapore
ygwen@ntu.edu.sg, sunp0003@e.ntu.edu.sg, asjfcai@ntu.edu.sg

Abstract

User-generated content (UGC) is touted as one of the driving forces in global media industry. However, due to its long-tail nature, UGC presents a list of design challenges in media distribution, which in turn dictate a set of required features to support UGC delivery. Existing solutions based on content delivery network (CDN), due to their lack of all the required features, are often inadequate. In this paper, we propose an innovative idea of Content-Delivery-as-a-Service (CoDaaS). Specifically, virtual Content Delivery Service (vCDS) is configured, in an on-demand manner, for UGC providers to distribute their contents to a group of designated consumers. The proposed CoDaaS solution builds on a hybrid media cloud, and offers elastic private virtual content delivery services with an agreed Quality of Service (QoS) to UGC providers. We have built a system prototype, complemented with a simulation platform, with an objective to establish technical feasibility and optimize system performance. The preliminary results, obtained from two pilot applications, validate all the required features for UGC delivery and verify its comparative performance advantages. Further research will focus on extending the prototype into an open-access platform for application development, thus accelerating the innovations in UGC delivery.

Keywords: Virtual content delivery service, Cloud computing, User generated content, Content distribution network, Content-Delivery-as-a-Service.

1 Introduction

User-generated content (UGC), owing to the ubiquitous penetration of high-speed Internet and the advancements in media technologies, is emerging as one of the dominant forms of global media. UGC refers to media contents (e.g., photos, videos and podcasts), which are typically created by nonprofessional producers (e.g., individuals, nonprofit organizations, and small-to-medium businesses). Serving both personal and business purposes, UGC is currently one of the fastest growing forms of contents on the Internet [2]. As shown in Figure 1, in 2008, 42.8% of Internet users in the USA (i.e., 82.5 million) contributed to some form of UGC, and that ratio is expected to reach 51.8% by 2013. This dramatic growth of UGC is expected to contribute significantly to the exponential growth of Internet traffic [1]; thus efficient mechanisms should be sought to distribute UGC, preferably at a low cost.

However, UGC, due to its long-tail nature [3] and the unique requirements of UGC providers, posits additional challenges in its delivery. The inherent long-tail characteristic of UGC, that is, the majority of UGCs are of low interest for the general public, contradicts to how the prominent content delivery network (CDN) operates. The leading CDN service providers (e.g., Akamai, Limelight, etc), tailor their network architecture and operational structure toward popular contents. As a result, it is difficult to distribute heavily long-tailed contents over CDNs in a profitable way. Moreover, unique requirements of UGC delivery, for example, micro-transaction, personalized QoS, and a limited set of consumers (cf Section 2), make the existing CDN-based solutions (cf Section 3), including CDN reseller, Content Broker and Cloud CDN, ineffective or inadequate.

In this paper, we propose a novel solution, Content-Delivery-as-a-Service (CoDaaS), by leveraging the emerging cloud computing technologies [6]. Our design objective is to distribute UGCs in the most economical fashion, while respecting the unique Quality-of-Service (QoS) requirements. Specifically, upon receiving a content-delivery request from a UGC provider[1], the (content-delivery) service provider dynamically calculates the most cost-effective content network overlay over an underlining media cloud and configures a private virtual content delivery service (vCDS) over the content distribution overlay. The resulting private content delivery service is then offered to the content provider, in a pay-per-use model, to distribute its UGCs via the content network overlay and render them in a context-aware fashion (e.g., network condition, capability of user devices, and location of consumers) to a list of specified consumers. Our proposed CoDaaS solution provides the required feature vector for UGC delivery and its performance advantages are verified via a proof-of-concept prototype system and a simulation platform. The paper is organized as follows. In Section 2, we first present the list of unique requirements of UGC delivery, which dictates a desirable feature vector. In

[1] In this paper, we use UGC provider, creator and contributor indistinguishably.

*Corresponding author: Yonggang Wen; E-mail: ygwen@ntu.edu.sg
DOI: 10.6138/JIT.2013.14.3.01

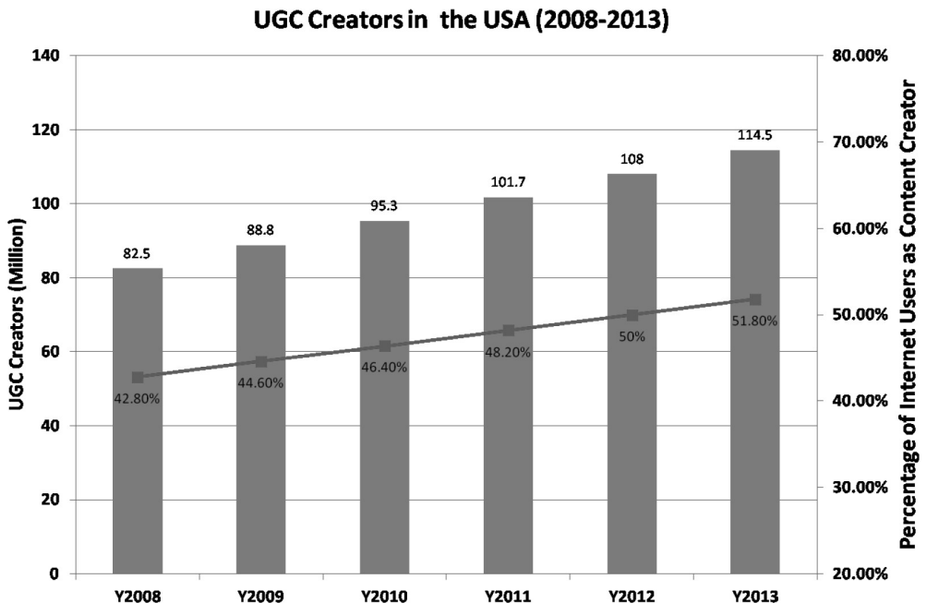


Figure 1 Growth of User-Generated Content Creators in the USA from 2008 to 2013

Source: Adapted from [2].

Section 3, we first present a survey of potential candidate solutions, based on CDN architecture, and then follow up with an overview of our proposed CoDaaS solution. In Section 4, we present a detailed architecture for the CoDaaS solution with its building modules. In Section 5, we present an implementation of our proposed CoDaaS platform in a private cloud environment, and its performances in two pilot application (i.e., file distribution and media streaming) are found to outperform a benchmark case. Section 6 summarizes this paper.

2 Characteristics and Features for UGC Delivery

In this section, we first identify a list of design requirements of UGC delivery and then propose a feature vector to meet these requirements. Specifically, a mapping between requirements[2] and features, listed in Table 1, is elaborated as follows:

- Micro-Transaction: The traffic volume from an individual UGC contributor is normally small, and varies significantly over the time. As a result, to avoid wasting system resources, delivery service cannot be configured statically, as in CDN-based solutions. The supporting feature for this requirement is an elastic delivery service, in which resources for content delivery can be allocated dynamically to support the desired quality of service efficiently.
- Personalized QoS: UGC contributors, when distributing their contents, often require a higher-level control of the delivery process. This control is manifested through a list of elevated QoS metrics. For example, for business-crucial contents, the provider requires the delivery service with a high level of security and privacy. Therefore, any solution should provide the capability of personalizing its content delivery service, thus providing private delivery service.
- Social Group Management: The consumers for UGCs are normally friends or customers of their contributors, often forming a tightly-knitted social group. Moreover, with the wide adoption of social networking (e.g., Facebook and Twitter), more and more UGC providers will distribute their contents to groups or subgroups in existing social networking. The supporting feature for this requirement is a social group management function, which can integrate contributors' private social groups with popular social networking providers.
- Context-Aware Consumption: One of the key reasons for UGC delivery is to provide the content consumers with a better media experience. Context-aware media consumption stands out as one top candidate to meet this requirement. Specifically, media consumption should adapt to real-time network status (e.g., bandwidth, fading, delay, etc), diverse capabilities of user devices, and even the location of content consumers. The supporting feature is an *adaptive media processing/rendering function* at the ingress point and the egress point of the virtual content delivery service.

[2] This set of design requirements was based on a private market study by one of the biggest companies in networking equipments.

Table 1 UGC Delivery Characteristics, Feature Vector and Potential Solutions

Requirements	Features	CDN Reseller	Content Broker	Cloud CDN	CoDaaS
Micro-Transaction	Elastic Delivery Service	Yes	No	Yes	Yes
Personalized QoS	Private Delivery Service	Yes	No	Yes/No	Yes
Social Group Management	Social Networking Function	No	Yes	No	Yes
Context-Aware Consumption	Adaptive Media Rendering Function	No	No	No	Yes
Technology-Unsavvy User	Turn-Key Solution	No	Yes	No	Yes
Low Cost	Pay-Per-Use Model	Yes/No	Free	Yes	Yes

- Technology-Unsavvy User: The UGC contributors are often technology-unsavvy users. As a result, any delivery solution should not require its users to follow complicated steps to configure the service, for example, changing its URL [15] or redirecting its DNS to a proxy. To support this group of users, one would need a *turn-key solution* that requires, if not no, a minimum reconfiguration.
- Low Cost: The UGC delivery service is designed for a consumer market, in which clients are highly sensitive to price. The cost of the service should be minimized by globally optimizing system utilization and the consumption of the service should be based on a pay-per-use model, instead of a subscription model. The supporting feature for this requirement, from an end user's perspective, would be a *pay-per-use model*.

3 Candidate Solutions for UGC Delivery

In this section, we first outline a few potential CDN-based solutions for UGC delivery. However, none of these CDN-based solutions can support all the required features for UGC delivery, and thus they are inadequate. To address this feature gap, we propose a novel approach, i.e., Content-Delivery-as-a-Service (CoDaaS).

3.1 CDN-Based Solutions

One family of solutions for UGC delivery is to extend commercial CDNs into a retailer mode for UGC delivery. A CDN consists of a pool of distributed servers, each of which contains copies of data and all of which collaboratively serve all the users. The CDN is designed to improve access to the data it caches by increasing access bandwidth and redundancy and reducing access latency. Previously, CDN has been a huge success to deliver popular contents. Leading CDN providers include Akamai, CDNetworks, EdgeCast Networks, Intermap, Level-3 Communications, Limelight Networks, NetDNA, etc.

However, directly applying CDN for UGC delivery is challenging. First, it has been known that the average cost of CDN service is prohibitively high [4], and thus out of reach for many small-to-medium business (SMB), non-profit organizations or individuals. Second, CDN providers, due to its operational model for popular contents, are unwilling to deal with a large number of small players in the market. As a result, CDN cannot be applied for UGC delivery directly. Nevertheless, the capabilities of CDN, for example, global coverage and reliable technologies, are attractive for UGC delivery. As such, commercial solutions have been exploring CDN's capabilities for UGC delivery.

Currently, CDN-based solutions that are potentially suitable for UGC delivery can be classified into three categories, including CDN reseller, Content broker and Cloud CDN. Details of these solutions are as follows:

(1) *CDN Reseller*: In this solution, a third-party service provider first contract a volume CDN service from an original CDN provider (e.g., Akamai, Level-3 Communications, etc.), and then resell the CDN service in small pieces to clients with a low demand and/or a short duration. A partial list of the CDN resellers include:

- DistributionCloud: It leverages Akamai's CDN service to provide a cloud-based site distribution service. Specifically, it uses Akamai's EdgePlatform Network to optimize routes and replicate data dynamically to deliver contents and applications more quickly, reliably and securely. Its service is offered via a subscription model.
- VPS.net: It uses the CDN service from Level-3 communications to support its site distribution service. Specifically, content providers can rent CDN nodes to deliver their static files worldwide and put them closer to the end users. Its service is offered via a subscription model.
- Brightbox CDN: It is powered by CDNetworks to provide its customers with an easy, scalable and reliable way to dramatically accelerate performance of their web applications. Its service works by seamlessly delivering static contents (e.g., images, css and javascript) to the end users from a high performance edge cache server nearest to their geographic location. It offers both a "pay-as-you-go" option and a 12-month commit option for clients.
- Rackspace: It leverages Akamai's CDN service to power its CloudFile service, which provides unlimited online storage for files and media, and delivers these contents

to consumers at a blazing speed over Akamai's CDN. Its service is charged on a pay-per-use model, in which the clients only pay for the file storage space and outgoing bandwidth that the service actually uses.

As explained, the content delivery service offered by CDN resellers is elastic and private. However, because their services are designed typically for site distribution, they do not provide native support to the social networking function and the adaptive media processing/rendering function. Moreover, clients have to create a dedicated website for content distribution, which could be a technology barrier for its wide adoption to technology-unsavvy clients. The service can be consumed with a pay-per-use model, but not for all the providers.

(2) *Content Broker*: In this solution, UGC providers upload their contents to a third-party website (e.g., YouTube, Facebook, etc.), which serves as a central exchange place for end users to consume these contents. The website either has its own in-house CDN or outsources content delivery through a long-term contract with CDN providers. The advantages of the content broker solution include: (i) it has a native support to social networking function, and (ii) it requires almost no configuration from the clients. However, it does not provide elastic and private delivery services, because all the contents are aggregated together for delivery. Moreover, it normally does not provide adaptive media processing/rendering capabilities, e.g., YouTube limits the format of uploaded contents. Its service, normally sponsored by ad revenue, is free.

(3) *Cloud CDN*: An emerging trend is for cloud service providers to sell CDN services. Two leading cloud CDN service providers are outlined here, including:

- CloudFront: Amazon CloudFront is a web service for content delivery. It integrates with other Amazon Web Services to give developers and businesses an easy way to distribute contents to end users with low latency, high data transfer speeds, and no commitments.
- CloudFlare: CloudFlare protects and accelerates any website online, by routing contents via its intelligent global network. It automatically optimizes the delivery of webpages toward the fastest page loading time and best performance.

Due to its cloud nature, the cloud CDN service is elastic and usually adopts a pay-per-use model. CloudFlare provides a private delivery service. However, to work with these cloud CDN service providers, clients usually follow complicated steps to configure their own servers. For example, CloudFront needs to map the client's domain name into a distribution domain name. Moreover, it normally does not provide native support for social networking management.

Table 1 captures a feature matrix for these CDN-based solutions. As mentioned previously, these CDN-based solutions cannot meet all the requirements for UGC delivery. The problem is similar to the small-transaction problem in the banking system. Traditional banking system works well for large transactions, while they are inadequate for small transactions, which are more popular on the Internet. It motivated innovative solutions (e.g., Paypal and Alipay). Likewise, for content delivery, UGC motivates us to develop novel solutions tailored for its specific requirements, as explained in next subsection.

3.2 Content-Delivery-as-Service (CoDaaS)

To support all the requirements of UGC delivery, we propose the idea of Content-Delivery-as-a-Service (CoDaaS). CoDaaS leverages cloud computing technologies (e.g., resource virtualization technology [5]), with an objective to provide elastic, private content delivery services, as a turn-key solution, to UGC providers, in a pay-per-use model. In addition, the solution provides a social networking function and context-aware media processing and rendering capabilities for better quality of experience (QoE).

An end-to-end view of CoDaaS is illustrated in Figure 2, over a hybrid media cloud infrastructure. The media cloud consists of a set of interconnected data centers, forming an un-configured content-distribution overlay with an underlying network infrastructure. Each media data center provides a shared pool of IT resources, including computing, storage and bandwidth, to media service providers via resource virtualization techniques [7]. For example, physical servers, enabled by supervisors (e.g., VMware Server, Xen, Microsoft Hyper-V, etc.), can be abstracted into a pool of virtual machines (VM). Each VM is an instantiation of a logical server, which runs its own guest operating system and media applications (e.g., streaming engines or transcoding engines). Network can

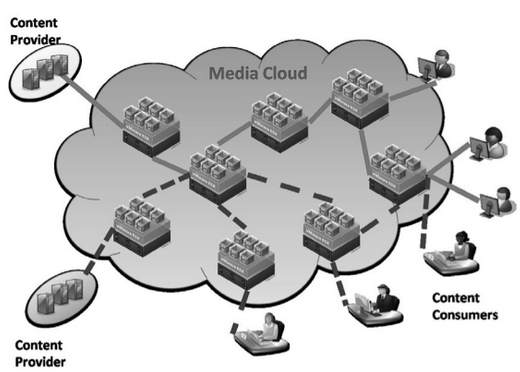


Figure 2 An End-to-End View of CoDaaS: It Dynamically Rchestrates and Configures Virtual Content Delivery Service Overlays for UGC Delivery

also be virtualized with existing VPN solutions or via emerging OpenFlow technologies [8]. CoDaaS operates as follows.

First, UGC providers, when publishing their contents, submit a content-delivery request to a media service provider. Each request is specified with four components:

(1) a list of *content locations*, which could be their private servers or storage space rent from public cloud service providers (e.g., Amazon S3, Dropbox);
(2) a group of *content consumers*, which could be a group of friends on a specific social networking platform or all the users in some geological location or within some subnet;
(3) a list of desired *QoS metrics* (e.g., bandwidth, delay, time jitter, security, and etc), and
(4) a *contract duration* during which contents are available.

These components are specified in a standard service level agreement (SLA), and encoded in an XML file.

Second, when the media service provider receives a content-delivery request, a virtual content delivery service (vCDS) overlay is carved out of the underlying media cloud. Specifically, the vCDS overlay consists of a list of virtual machines (VM) with allocated storage space, interconnected with virtualized network connections. The content distribution topology is formed with two objectives: (i) it provides the required QoS, and (ii) it has a minimum (monetary) cost. In addition to the standard storage and bandwidth required by content delivery, VM provides computing power for content processing/rendering functionalities (e.g., media transcoding [9]) and managerial functionalities (e.g., content management). The amount of IT resources allocated to the delivery service can scale up and down, meeting its application demand.

Third, when the virtual content delivery service is configured, contents are acquired from their storage locations into the CoDaaS system and optimally pushed into a set of media engines, from which the targeted audience can consume them with an enhanced QoE. Moreover, contents from UGC providers are encrypted with public keys before the metadata and data associated with contents are distributed among CoDaaS storage nodes. The keys with UGC are revokable, which gives the content provider the ultimate control of their content delivery.

Finally, when the content-delivery contract expires, the media service provider tears down the virtual content service by returning the required IT resources back to the resource pool.

In this research, a virtual content delivery service (vCDS) is manifested through a vCDS profile, which is the combination of client requests and resources reserved/configured for content distribution. In particular, resources can be classified into two categories, including:

- IT resources including computing power, storage space and network bandwidth;
- Application resources including individual services for social group, content metadata, etc.

To support the virtual content delivery service, we present a system functional specification for our proposed CoDaaS in next section.

4 System Functional Architecture for CoDaaS

In this section, we present a system functional architecture for CoDaaS, with a detailed description of all the modules.

Figure 3 illustrates a system functional diagram. It consists of three main functional layers. A cloud infrastructure layer and a content service virtualization layer are integrated through an inter-cloud messaging bus, which is a distributed protocol for resource management, account management, content management, service orchestration, and etc.

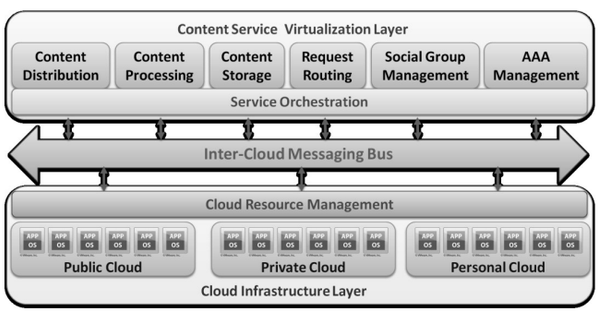


Figure 3 System Functional Architecture for CoDaaS: It Embodies Three Alternative Cloud Paradigms, Including Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS)

4.1 Cloud Infrastructure Layer

In this layer, IT resources are carved from a hybrid media cloud (e.g., public cloud, private cloud and personal cloud) to form a dynamic content delivery service overlay. The content delivery service overlay consists of both a content distribution network overlay for efficient media distribution, and a set of application engines, for media processing/rendering, managerial functions and service orchestration. The key module to pool all the required IT resources together is a distributed algorithm for cloud resource management.

4.2 Content Service Virtualization Layer

This layer provides a suite of middleware to support different features in UGC delivery. These components,

extended from a generic content delivery network in [10], include:

- Content Distribution: This module provides services for optimally distributing the content from its source to its consumers. These services are supported by content caches strategically located through the network. The distribution component also includes the actual mechanism and the protocols used for transmitting data over the underlined network and serving the contents to the consumers.
- Content Processing: This module provides services for acquiring or adapting contents to suit network conditions, user preferences and device capabilities. This includes modification or conversion of both contents and requests for contents. Examples include content adaption for wireless devices, or added privacy by making personal information embedded in user requests, or media transcoding into different formats when the content is injected into the CoDaaS platform from its provider.
- Content Storage: This module provides services for storing user-generated contents in the CoDaaS platform, in a security level specified by content providers. Example functionalities include distributed secure storage, content naming and resolution, content replications, and etc.
- Request Routing: This module provides services for navigating both clients' requests to upload their contents and consumers' requests to retrieve the contents, to the location that provides the best user experience. The selection of the most appropriate location is typically based on network proximity and availability of system resources. Therefore, a network monitoring module is also included in this component.
- AAA Management: This module deals with authorization, authentication and account, and provides services that enable monitoring, logging, accounting and billing of CoDaaS usage. Mechanisms in this module will ensure the identity and the privilege of all parties involved in a transaction, as well as, digital right management.
- Social Group Management: This module manages the communication within and across the social groups for CoDaaS applications, including both private social groups (e.g., GoogleTalk, client list, etc.) and public social groups from social networks (e.g., Facebook, Google+, etc.).
- Service Orchestration: This module provides mechanisms for integrating different media services, for both internal CoDaaS system and external service entry points, into an integrated media application for both content providers and content consumers. For example, a social TV application [11] would draw a lot of individual media services together, including a buddy service from a social networking, a media streaming service, to name a few.

4.3 Inter-Cloud Messaging Protocol

The inter-cloud messaging protocol is a distributed mechanism to connect all the participant components into an integrated media application. As presented in [12], the design of an inter-cloud messaging protocol can be decoupled into three design subtasks, including message content, message format and message vehicle.

(1) *Message Content*: It defines what information should be exchanged among different modules, to support CoDaaS. Based on our system architecture, message contents can be classified into the following domains:

- Content Distribution: messages for moving the content from its source to the users.
- Request Routing: messages for navigating requests to a location best suited for content uploading and retrieval.
- Content Processing: messages for acquiring or adapting content to suit network conditions, user preferences and device capabilities.
- Content Storage: messages for storing and searching contents in a distributed storage system.
- Service Orchestration: messages for orchestrating service mixture into media applications.
- Resource Management: messages for monitoring, allocating and configuring IT resources.
- Social Management: messages for social interactions among different groups.
- AAA Management: messages to enable monitoring, logging, accounting and billing of CoDaaS usage.
- Misc.: messages for extension purposes.

(2) *Message Format*: It refers to methods in which message contents are encoded to exchange among different participant components in CoDaaS. Three alternative message formats, including Type-Length-Value (TLV), Field-Value and XML, have been considered in our design. We choose to use the TLV format, because fast TLV parsers, for example, tlve, can be leveraged for fast message processing and the bit efficiency of TLV format is comparatively higher to reduce the message overhead.

(3) *Message Vehicle*: It refers to channels through which encoded messages are exchanged among different components. We have explored three alternative vehicles, including (i) RPC, (ii) Web Service, and (iii) Messaging Queue. In our design, we choose to deploy a message queuing bus, in particular, XMPP[3]. XMPP

[3] Extensible Messaging and Presence Protocol (XMPP) is an open-standard communication protocol for message-oriented middleware based on XML. Currently, XMPP is also being extended to handle signaling/negotiation for VoIP and other media sessions. It is designed to be consistent with the leading social networking platforms (e.g., the Google Talk service and the Facebook chat service) and bridgeable with the Session Initiation Protocol.

is widely supported in current network deployment and is considered as a top candidate for inter-cloud communication.

5 System Prototype and Performance Verification for CoDaaS

This section presents our proof-of-concept (PoC) system prototype and its performance in file downloading and media streaming. To validate all the proposed features in CoDaaS, we are developing a CoDaaS platform prototype, running over a private cloud infrastructure. In addition, a simulation platform is built for performance testing and design choice evaluation. Our preliminary performance results verify the superiority (in term of user experience) of our proposed solution.

5.1 PoC Demo System

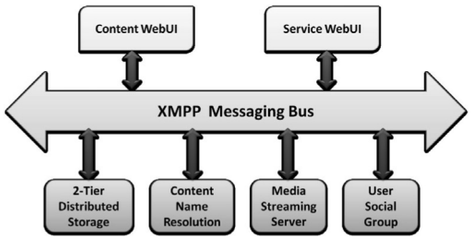
Our prototype system and simulation platform are based on our proposed functional architecture, as illustrated in Figure 3. In particular, we have implemented a few key components in the proposed architecture. Figure 4(a) highlights the set of implemented components for our PoC demo system. Details of each components are explained as follows:

(1) *Media Cloud Infrastructure*: In the layer of media cloud infrastructure, we aim to provide the minimum set of functionalities for UGC delivery. To achieve this objective, we have implemented four components in our testbed, including: (i) a 2-tier distributed file storage, (ii) a content naming and resolution mechanism, (iii) a media streaming server and (iv) a social group management module.

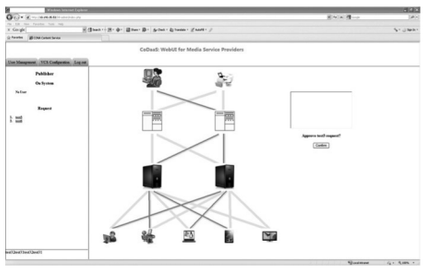
For the content storage system, we have developed a 2-tier file storage system. One tier of storage nodes locate at customers' premise; and the other tier of storage nodes locate at network edges. Contents are first stored at the first tier of storage nodes via an erasure-code based scheme, to improve data availability. In addition, contents are replicated at the second tier of storage nodes in network edge, to serve end-users with an improved user experience.

For the content naming and resolution mechanism, we have adopted a scheme similar to the one used in CCN/NDN [13-14]. Specifically, content is assigned with a unique URL name, which is encrypted with a public/private key system. An alternative approach would be to leverage the naming solution implemented in the Tonido system.

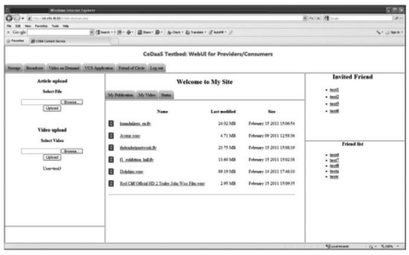
For the media streaming system, we have incorporated two alternative streamers, including the VLS media streamer and the Adobe HTTP dynamic streaming server. In the latter case, videos can be efficiently delivered to users by dynamically switching among different streams of varying quality and size during playback. This capability provides users with the best possible viewing experience that their bandwidth and local computer hardware can support, supporting the adaptive media consumption feature in UGC delivery.



(a) Software Architecture



(b) Content WebUI



(c) Service WebUI

Figure 4 Proof-of-Concept Prototype for CoDaaS: Back-End Software Architecture and Front-End User Interfaces

For the social group management, we have adopted an XMPP-based instant-messaging system. We have used the same database (i.e., MySQL) for both users and processes in the demo system. Intuitively, such a design choice allows us to treat all the identities (content, process, user, etc.) in the system in an equal basis, via the XMPP messaging bus.

(2) *Media Service Application*: In addition to the back-end system, we have also implemented in our testbed a set of interactive user interfaces for both content providers/consumers and service providers. In particular, we have developed two separate web user interfaces: one for content providers to submit media delivery service requests and content consumers to consume pushed contents, and the other for media service providers to monitor system status and configure personalized content delivery services.

Figure 4(b) illustrates the content WebUI for content providers and consumers. From left to right, it has the following parts: (i) a content management panel, (ii) a panel for media applications, and (iii) a user group panel. Moreover, the set of key functions it provides for content providers/consumers includes:

- Uploading contents to media clouds,
- Interacting with peers through social group plug-in,
- Submitting content delivery service request,
- Launching media applications.

Figure 4(c) presents the service WebUI for media service providers. Initially, our demo system only provides manual configuration. In our simulation platform, vCDS can be dynamically reconfigured automatically, according to user demand and resource availability. The ultimate goal is to configure vCDS in a self-adaptive fashion. The set of key functions it provides for media service providers include:

- Monitoring media cloud infrastructure,
- Configuring and bookkeeping virtual content delivery services,
- Visualizing real-time system dynamics.

With this PoC system, we have been able to demonstrate the required feature vector for UGC delivery and its performance advantages via two pilot applications, as detailed in next two subsections.

5.2 Feature Verification for UGC Delivery

In this subsection, we illustrate how the PoC demo system provides a feature vector that meets the unique requirements of UGC delivery.

The elastic and private delivery services are guaranteed by the on-demand configuration of the virtual content delivery service in the CoDaaS. Specifically, each virtual content delivery is uniquely assigned to a content request, with the required QoS. The social group management is built in with XMPP (in our system, the cross-platform OpenFire is used for this purpose). The adaptive media rendering function is provided by the Adobe HTTP dynamic streaming server; however, the adaptive media processing capability has not been included in this demo system. The web-based user interface is universal for both content providers and consumers, requiring almost zero configurations from the clients. The virtual content delivery service can be contracted in a pay-per-use model, minimizing the monetary cost for the content providers.

5.3 System Performance Validation

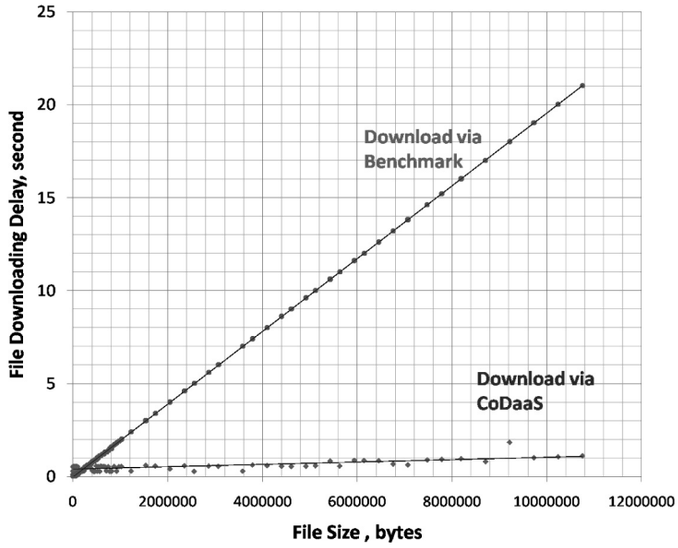
With the PoC system and the simulation platform, we have been able to conduct a few experiments to validate the performance advantages of CoDaaS. The benchmark scenario is an imaginary solution in which contents are distributed directly from client's home gateway. The benchmark is the worst case in UGC delivery and we are working on performance comparisons between CoDaaS and other CDN-based solutions (e.g., Content Broker).

(1) *File Sharing*: The application is for an individual to share a list of files from his/her home gateway, with a group of collaborators, in a secure manner. In the experiment, we randomly generate 100 content files, whose sizes range from 1K bytes to 10G bytes. Each file is originally stored at client's home gateway. Collaborators retrieve the list of shared files in two alternative ways. The first solution, also the benchmark case, is for them to download the content files directly from the home gateway. The second solution is to retrieve the list of files via CoDaaS. The end-to-end file downloading delay is used as a measurement of quality of service (or quality of experience) [16].

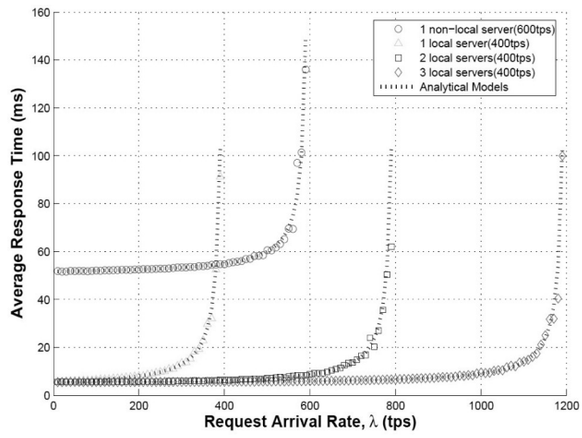
The file delay patterns under these two solutions are significantly different, as illustrated in Figure 5(a). In the benchmark case, the average throughput is around 500KBps which is limited by the uploading bandwidth of the home network; while in the CoDaaS solution, the average throughput ranges between 8MBps and 12 MBps, with an average of 10 MBps. In this simple application, our proposed CoDaaS solution has a throughput that is about 20 times higher than that of the benchmark case. The fundamental gain in this application comes from the time-shifting property of our CoDaaS solution, in which contents are prepositioned optimally to avoid the uploading bandwidth bottleneck. Notice that, when the file size is small, the benchmark case outperforms the CoDaaS solution. In this case, the downloading delay is dominated by the connection delay, depending on the capability of the server. In our testbed, the home gateway, which is a physical server, is more powerful than the edge server that is a VM; as such, the connection delay from the home gateway is smaller than that from the edge server.

(2) *Video Streaming*: In this experiment, we investigate the video streaming application for both the benchmark case and the CoDaaS solution. One QoS metric of interest is the response time, which is defined as the duration from the moment when the request for video is initiated and the moment when the video starts to play at consumer's device. In the benchmark case, video files are streamed from client's home gateway to a group of consumers. In this case, the end-to-end transport delay is found to vary between 40ms and 60ms, depending on ambient traffic. With the CoDaaS solution, all the video contents are pushed to up to three edge servers that are closer to the end users. The end-to-end delay between user and the edge server varies from 1 ms to 5 ms [17].

For the benchmark test, the home gateway is



(a) File Sharing



(b) Video Streaming

Figure 5 Performance Validation for CoDaaS: (a) File Retrieval Delay for File Sharing, and (b) Response Time for Video Streaming

configured to support 600 transaction per second (TPS), and the edge server with an average process rate of 400 TPS. We stress the testbed by increasing the request arrival rate (λ). Four cases were tested including: (i) one home gateway, (ii) one edge server, (iii) two edge servers, and (iv) three edge servers. Figure 5(b) plots the streaming response time under these four test scenarios, compared with the analytical results from an *m/m/K/K* queueing model, where the first K refers to the number of edge serves in the virtual media service profile, and the last K refers to the number of outstanding requests that one server can buffer. The buffer size depends on how the streaming server is implemented. In particular, the number is roughly equal to the total number of file descriptors (FD) one process can open (normally 64K in Linux) divided by the number of file descriptors allocated for each outstanding request. We set K = 10K in our experiment, emulating a streaming server that allocates six FDs for each incoming request (a portion of FDs are reserved for system usage). Notice that this number actually depends on the implementation of the streamer. It is obvious that the more number of edge servers is used, the better the performance is. At the same time, the total capability of edge server cluster should exceed the capability of the home gateway, especially when the system is close to its limit. The results verify that our CoDaaS solution provides better streaming response time for content consumers than the benchmark case.

6 Summary

In this paper, motivated by the ever-growing UGC market, we proposed an innovative solution, i.e., Content-Delivery-as-a-Service (CoDaaS), to provide private virtual content services to UGC providers. The solution is designed

to meet all the unique requirements of UGC delivery, for which existing commercial solutions based on content delivery network are inadequate. CoDaaS represents a paradigm shift from a wholesale content distribution market, as dominated by large CDN service providers (e.g., Akamai and Limelight), to a retail content delivery market.

This paper presents our proposed system architecture, with a PoC system prototype and a simulation platform. The PoC demo system verifies the desired feature vector for UGC delivery and its scalable performance. The simulation platform allows us to test the performance under different system parameters and operational algorithms, and the resulted insights can be used to develop an open-access platform for application development.

The performance presented in this paper is preliminary. In-depth performance optimization is being conducted on the PoC demo system and the simulation platform. In particular, we are optimizing the system performance with difficult operational algorithms, including collaborative caching algorithm, user redirection algorithm [18], distribution overlay algorithm with different cost metrics, etc. The ultimate objective is to push the system performance to the limit and understand the fundamental trade-off between the cost and the quality-of-service for UGC delivery.

Acknowledgements

The authors would like to thank beneficial discussions with Prof. Dah-Ming Chiu from the Chinese University of Hong Kong, Dr. Haiyong Xie, Dr. Xinwen Zhang, Dr. Wei Jun at Huawei R&D Center, North America. In addition, the corresponding author would like to thank the generous support from NTU start-up grant.

References

[1] *Cisco visual networking index: Global mobile data traffic forecast update, 2010-2015*, February, 2011. Cisco White Paper.

[2] Paul Verna, *A Spotlight on UGC Participants*, 2009, http://www.emarketer.com/Article/Spotlight-on-UGC-Participants/1006914

[3] Meeyoung Cha, Haewoon Kwak, Pablo Rodriguez, Yong-Yeol Ahn and Sue Moon, *Analyzing the Video Popularity Characteristics of Large-Scale User Generated Content System*, *IEEE/ACM Transactions on Networking*, Vol.17, No.5, 2009, pp.1357-1370.

[4] Jussi Kangasharju, James Roberts and Keith W. Ross, *Object Replication Strategies in Content Distribution Networks*, *Computer Communication*, Vol.25, No.4, 2002, pp.367-383.

[5] Efraim Turban, David King, Judy McKay, Peter Marshall, Jae Lee and Dennis Viehland, *Electronic Commerce 2008: A Managerial Perspective* (5th ed.), Prentice Hall, Upper Saddle River, NJ, 2007, p.27.

[6] Peter Mell and Timothy Grance, *The NIST definition of cloud computing (draft)*, January, 2011. NIST Special Publication 800-145 (Draft).

[7] Paul Barham, Boris Dragovic, Keir Fraser, Steven Hand, Tim Harris, Alex Ho, Rolf Neugebauer, Ian Pratt and Andrew Warfield, *Xen and the Art of Virtualization*, *ACM SIGOPS Operating Systems Review*, Vol.37, No.5, 2003, pp.164-177.

[8] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker and Jonathan Turner, *OpenFlow: Enabling Innovation in Campus Networks*, *ACM SIGCOMM Computer Communication Review*, Vol.38, No.1, 2008, pp.69-74.

[9] Bo Shen, Sung-Ju Lee and Sujoy Basu, *Caching Strategies in Transcoding-Enabled Proxy Systems for Streaming Media Distribution Networks*, *IEEE Transaction on Multimedia*, Vol.6, No.2, 2004, pp.375-386.

[10] Markus Hofmann and Leland Beaumont, *Content Networking: Architecture, Protocols and Practice*, Morgan Kaufmann, San Francisco, CA, 2005.

[11] Pablo Cesar and David Geerts, *Past, Present and Future of Social TV: A Categorization*, *Proc. of IEEE Consumer Communications and Networking Conference*, Las Vegas, NV, January, 2011, pp.347-351.

[12] Yonggang Wen, Guangyu Shi and Guoqiang Wang, *Designing an Inter-cloud Messaging Protocol for Content Distribution as a Service (CoDaas) over Future Internet*, *Proc. of the 6th International Conference on Future Internet Technologies*, Seoul, Korea, January, 2011, pp.91-93.

[13] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs and Rebecca L. Braynard, *Networking Named Content*, *Proc. of CoNEXT 2009*, Rome, Italy, December, 2009, pp.1-12.

[14] Michael Meisel, Vasileios Pappas and Lixia Zhang, *Ad Hoc Networking via Named Data*, *Proceedings of the Fifth ACM International Workshop on Mobility in the Evolving Internet Architecture*, Chicago, IL, September, 2010, pp.3-8.

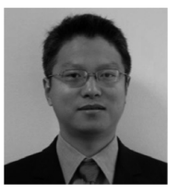
[15] Michael J. Freedman, *Experiences with CoralCDN: A Five-Year Operational Review*, *Proceedings of 7th USENIX/ACM Symposium on Networked System Design and Implementation (NSDI'10)*, San Jose, CA, May, 2010, p.7.

[16] Chin-Feng Lai, Honggang Wang, Han-Chieh Chao and Guofang Nan, *A Network and Device Aware QoS Approach for Cloud-Based Mobile Streaming*, *IEEE Transactions on Multimedia*, Vol.99, 2013, doi:10.1109/TMM.2013.2240270.

[17] Chin-Feng Lai, Sung-Yen Chang, Yueh-Min Huang, Jong Hyuk Park and Han-Chieh Chao, *A Portable UPnP-Based High Performance Content Sharing System for Supporting Multimedia Devices*, *Journal of Supercomputing*, Vol.55, No.2, 2011, pp.269-283.

[18] Yichao Jin and Yonggang Wen, *Content Routing and Lookup Schemes Using Global Bloom Filter for Content-Delivery-as-a-Service*, *Proc. of 2012 IEEE Global Communications Conference*, California, CA, December, 2012, http://www.ntu.edu.sg/home/ygwen/Paper/JW-Globecom-12.pdf

Biographies



Yonggang Wen (S'99-M'08) received his PhD degree in Electrical Engineering and Computer Science (with minor in Western Literature) from Massachusetts Institute of Technology. He is currently an assistant professor with School of Computer Engineering at Nanyang Technological University, Singapore. Previously, he has worked in Cisco as a Senior Software Engineer for content networking products. He has also worked as a Research Intern at Bell Laboratories, Sycamore Networks and Mitsubishi Electric Research Laboratory (MERL). He has published more than 50 papers in top journals and prestigious conferences. His system research on Cloud Social TV has been featured by international media (e.g., The Straits Times, The Business Times, Lianhe Zaobao, Channel News Asia, ZDNet, CNet, United Press International, ACM Tech News, Times of India, Yahoo News, etc.). His research interests include cloud computing, mobile computing, multimedia network, cyber security and green ICT. Dr. Wen is a member of Sigma Xi (the Scientific Research Society), IEEE and SIAM.

Peng Sun received his Bachelor's degree in automation engineering from Shan Dong University (SDU) in 2012. He is currently a PhD student in the School of Computer Engineering at Nanyang Technological University (NTU) in Singapore. His research interests include cloud computing, named data network and parallel computing.

Jianfei Cai (S'98-M'02-SM'07) received his PhD degree from the University of Missouri-Columbia. Currently, he is an associate professor and also the Head of Visual & Interactive Computing Division at the School of Computer Engineering, Nanyang Technological University, Singapore. His major research interests include visual signal processing and multimedia networking. He has published more than 100 technical papers in international conferences and journals. He has been actively participating in program committees of various conferences. He serves as the leading Technical Program Co-Chair for IEEE International Conference on Multimedia & Expo (ICME) 2012 and the leading General Co-Chair for Pacific-rim Conference on Multimedia (PCM) 2012. He was an invited speaker for the first IEEE Signal Processing Society Summer School on 3D and high definition / high contrast video process systems in 2011. He is also an associate editor for *IEEE Transactions on Circuits and Systems for Video Technology (T-CSVT)*, and a senior member of IEEE.