Designing an Inter-Cloud Messaging Protocol for Content Distribution as a Service (CoDaaS) over Future Internet

Yonggang Wen
School of Computer Engineering
Nanyang Technological University
Nanyang Avenue, Singapore 639798
+65 6790 6562
ygwen@ntu.edu.sg

Guangyu Shi, Guoqiang Wang
Innovation Center, Huawei
2330 Central Expressway
Santa Clara, CA USA 95050
(1)408 330 4563
{shiguangyu, gq.wang}@huawei.com

ABSTRACT

User-generated content (UGC) has become one of the dominant media formats. However, current approach to distribute UGC adds two middlemen (e.g., content aggregator and content distributor) in the value chain, resulting in a higher cost for media consumers. We have proposed a new model for content providers to request content-distribution-as-a-service (CoDaaS) in an on-demand fashion from service providers, thus bypassing the middlemen and reducing the total cost of content distribution. In this paper, we extend the concept of CoDaaS over different flavors of future Internet architecture. One challenge to provide such an emerging service over different Internet architecture is to design an inter-cloud messaging protocol to provide interoperability among all participant network clouds. To tackle this challenge, we structure the inter-cloud messaging protocol into three components including message vehicle, message format and message content. In particular, we highlight a few options for message vehicle and formats and offer our initial choice with rationale; we also categorize message contents into six sub-domains. This structural approach would make our work toward a proof-of-concept system easier.

Categories and Subject Descriptors

C.2 [**Computer-Communication Networks**] Network Architecture and Design, Distributed Systems

General Terms

Algorithms, Performance, Reliability, Security

Keywords

Cloud Computing, Content-Delivery-as-a-Service, User-Generated Content

1. INTRODUCTION

User-generated content (UGC) has become one of the dominant forms of global media, resulting in a tremendous growth opportunity [1]. However, current approach to distribute UGC, consisting of content aggregators (e.g., Youtube and Facebook) and content distributors (e.g., Akamai and Limelight), does not serve well with the ever-growing number of UGC providers. First, the existing solution cannot deliver contents to consumers with a vector of quality-of-service (QoS) metrics, specified by content providers. Second, the cost for UGC providers to contract content delivery with distributors is prohibitively high. To address this technology and market gap, we have proposed in [2] a new model for content providers to request content-distribution-as-a-service elastically and directly from service providers. It is our belief that our proposal entails a solid technology to shortcut the middlemen in content distribution value chain and reduces the total cost for content distribution.

Our previous paper focuses on developing CoDaaS over existing TCP/IP network, to capitalize on the UGC growth opportunity. In this paper, we propose to extend CoDaaS as an emerging content service over future Internet. However, it is still too early to decide which candidate architecture will prevail in future. In light of the ongoing research on future Internet architecture [3-6], we choose to integrate CoDaaS with a list of the most promising future Internet architectures. Indeed, since CoDaaS can be simply considered as an emerging content service that can be decoupled from the underlying transport network, it is technically possible to run a large-scale CoDaaS overlay over a hybrid cloud of different networks. One of the key challenges in this situation is to design a scalable and robust inter-cloud messaging protocol for CoDaaS to spread across different transport networks.

In this paper, we highlight our thought process in architecting a scalable and robust inter-cloud messaging protocol. In particular, we structure the protocol into three components: i) message vehicle for transporting messages among different clouds, ii) message format that encodes message, and iii) message content that carries information. In each component, we evaluate a few candidate solutions and offer our initial choice of design. We are working on developing a prototype of our proposed messaging protocol at NTU.

The paper is organized as follows. In Section 2, we present our proposed CoDaaS system architecture. In Section 3, we present our thought process in designing the inter-cloud messaging protocol, which is structured into three components. Section 4 summarizes the paper.

2. CoDaaS System Architecture

In this section, an end-to-end description of how CoDaaS works is presented. Since CoDaaS is positioned as a novel content service for future content networking, it is expected that it will work with different flavors of future Internet architecture. In particular, CoDaaS could run over a hybrid cloud of different Internet

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CFI'11, June 13–15, 2011, Seoul, Korea.
Copyright 2011 ACM 978-1-4503-0821-2/11/06…$10.00.

architectures. As such, an inter-cloud messaging protocol is highly desired to provide interoperability.

2.1 CoDaaS Concept

In this subsection, we recapture our proposed CoDaaS [2], which provides content distribution service for UGC providers to deliver their contents to a list of specified consumers, with a specific service level agreement (e.g., QoS metrics).

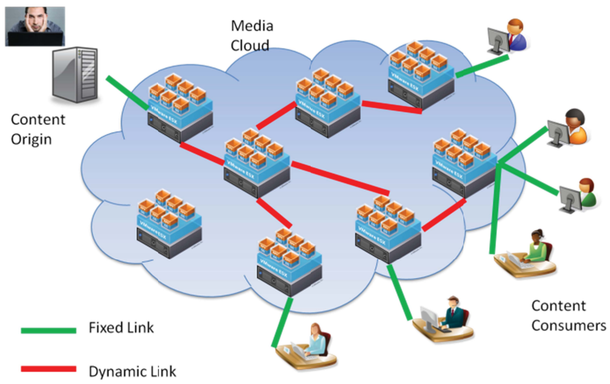

Figure 1. Content-Delivery-as-a-Service (CoDaaS) over a hybrid media cloud infrastructure

In Figure 1, we illustrate our proposed CoDaaS architecture over a hybrid media cloud infrastructure. The media cloud consists of a list of interconnected data centers or application routers, forming a content-distribution-network (CDN) overlay over data transport network. UGC providers, when publishing their contents, submit a content-delivery request to some service provider. Each content-delivery request includes four components: i) a list of locations for content origins, ii) a list of targeted consumers, iii) a vector of QoS metrics, and iv) a time window during which the content should be published. These components are specified in a standard service level agreement (SLA) [7], which is normally encoded in a XML file. When service provider receives a content-delivery request, a virtual CDN overlay is carved out of the underlying media cloud. Specifically, the CDN overlay consists of a list of virtual machines (VM) that form a content distribution tree. The content distribution tree is calculated with two objectives: i) it has to provide the required QoS, and ii) it has a minimum cost. When the content-delivery window expires, the service provider tears down the content delivery service by returning the required IT resources back to the resource pool.

2.2 Models for CoDaaS over Future Internet

CoDaaS can be considered as a novel media service for future content networking. Thus, it is expected for it to run over different future Internet architectures. In this subsection, we outline two alternative models for CoDaaS running over a set of different network clouds, based on its signaling mechanism.

In Figure 2, two alternative models are proposed for CoDaaS running over a set of different network clouds, including:

a) Federated Model: in this model, all participant network clouds exchange control information via a federator, which could be controlled by a third-party entity or be embedded in CoDaaS.
b) Peering Model: in this model, all participant network clouds exchange control information without a federator, but through a shared message channel.

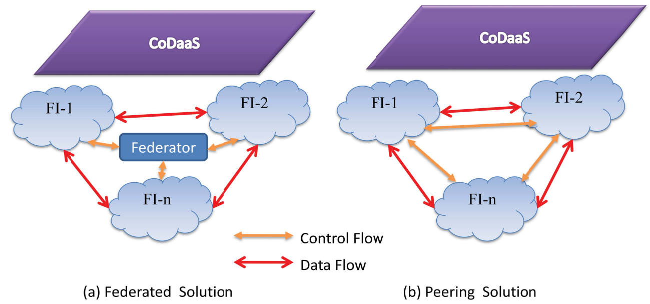

Figure 2. Internetworking Mechanism for Content-Delivery-as-a-Service over a set of different network clouds: a) federated model, b) peering model

These two models have their own pros and cons. The federated model provides good manageability, but it is hard to scale and suffers from single point of failure. The peering model provides good scalability, but it is hard to manage a distributed messaging channel. In our prototype, for high reliability, we choose to work with the peering model by designing a scalable and robust inter-cloud message protocol, as detailed in next section.

3. Inter-Cloud Messaging Protocol

In this subsection, we follow a "divide-and-conquer" strategy to structure designing an inter-cloud messaging protocol into three cascaded modules, including i) message content, ii) message format and iii) message vehicle, as shown in Figure 3. Each of these components is detailed in next three subsections.

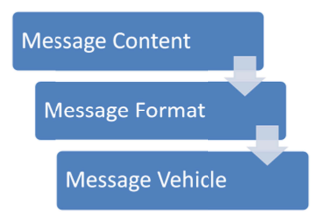

Figure 3. Breakdown of designing an inter-cloud message protocol

3.1 Message Content

In the first module, we need to define what information should be exchanged among different network clouds, to support CoDaaS. We refer the set of information to message content. Inspired by CDN solution [8], message contents can be classified into the following six domains:

1) *Content distribution*: messages for moving the content from its source to the users.
2) *Request routing*: messages for navigating user requests to a location best suited for retrieving the requested content.
3) *Content rendering*: messages for creating or adapting content to suit user preferences and device capabilities.
4) *Resource management*: messages for monitoring, allocating and reserving resources.
5) *Authorization, authentication and accounting*: messages to enable monitoring, logging, accounting, and billing of content usage.
6) *Misc*: messages for extension purposes.

The set of detailed messages are under development at NTU, and will be presented to research community when it is available.

3.2 Message Format

In the second module, message contents are encoded into different formats. Message format refers to methods in which message are encoded to exchange among different participant network clouds. In this design, we consider two alternative message formats, including:

1) Type-Length-Value: in the format, message is encoded as a type-length-value (TLV) element. Type is normally a binary code that indicates the category represented by this part of the message. Length is the size of the value field (typically in bytes or words). Value is a variable-size series of bytes that contains data for this part of the message. This format is popular in TCP/IP protocols.
2) Field-Value: in this format, messages are encoded as text-based “Field: Value” pairs, formatted according to RFC 2822 [9]. This format is commonly used in TCP/IP-based protocols, such as, HTTP, FTP, SMTP, POP3, and SIP.
3) XML: in this format, XML is used to implement messaging between different nodes in the network clouds. These messages are typically fixed with line-based text commands, such as with BEEP [10].

In our work, we choose to exchange messages in TLV format, because fast TLV parser, for example, tlve [11], can be leveraged for fast message processing, and the bit efficiency of TLV format is high to reduce messaging overhead.

3.3 Message Vehicle

In the third module, encoded messages are exchanged through some channel among different network clouds. Message vehicle refers channels through which messages are exchanged among different participant network clouds. In our design, we consider three candidate message vehicles, including:

1) RPC: in this design, each network cloud is assumed to have a portal server running an interchange process or daemon. These interchange daemons are responsible for inter-cloud communication and messages are exchanged through remote procedural calls among these daemons.
2) Web Service: in this design, all the participant network clouds provide a web service entry point for message exchange. Messages can be carried over standard protocols, such as, SOAP [12]. One way to implement it is to define the interfaces in a WSDL file and allow each network clouds to implement their own internal logic.
3) Messaging Queue: in this design, we leverage a distributed message queuing bus across all participant network clouds. Examples of scalable and robust message queuing bus include XMPP [13] and AMQP [14].

In our work, we commit to the message queuing bus, in particular, XMPP, because it is widely supported in current network deployment and is considered as a top candidate for inter-cloud communication [15]. Moreover, in XMPP, we can use the same database for contents, processes and users, which makes it easier to integrate with future name-data network.

4. Summary

In this paper, we share our thought process in designing an inter-cloud messaging protocol for our proposed CoDaaS to integrate with a list of promising future Internet architecture. In particular, we structure the designing process into three cascaded modules, including i) message contents, ii) message format and iii) message vehicle. For message contents, we design a taxonomy method to classify information into six domains. For message format and vehicle, we evaluate a few design options first and commit to one design choice. As our prototype work progresses, these decision choices will be revisit based on lessons learnt in the process.

5. ACKNOWLEDGMENTS

Our thanks go to NTU for their startup grant to support this research.

6. REFERENCES

[1] Paul Verna, 2007, User-Generated Content: Will Web 2.0 Pay Its Way? *eMarketer Report* , DOI= http://www.emarketer.com/Report.aspx?code=emarketer_2000421 .

[2] Wen, Y.G., Shi, G.Y. and Wang, G.Q. Content-Distribution-as-a-Service (CoDaaS): Case in Point of Designing Virtual Content Service Overlay, submitted to ACM SIGCOMM Workshop on Information-Centric Networking (ICN-2011).

[3] Jacobson, V.; Smetters, D. K.; Thornton, J. D.; Plass, M. F.; Briggs, N.; Braynard, R. Networking named content. Proceedings of the 5th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT 2009); 2009 December 1-4; Rome, Italy. NY: ACM; 2009; 1-12.

[4] Christian Dannewitz, NetInf: An Information-Centric Design for the Future Internet, Proc. 3rd GI/ITG KuVS Workshop on The Future Internet, May 2009.

[5] NDN, DOI = http://www.named-data.net/.

[6] European Future Internet, DOI = http://www.future-internet.eu .

[7] An outline of the core elements of an SLA, DOI = http://www.sla-zone.co.uk/

[8] Hofmann, M. and Beaumont, L.R. Content Networking: Architecture, Protocols and Practice, 2005.

[9] RFC 2822, Internet Message Format. DOI= http://www.ietf.org/rfc/rfc2822.txt.

[10] RFC 3080, The Blocks Extensible Exchange Protocol Core. DOI= http://tools.ietf.org/html/rfc3080.

[11] Open source package, DOI= http://tlve.sourceforge.net/.

[12] SOAP, DOI= http://www.w3.org/TR/soap/.

[13] XMPP, DOI= http://www.xmpp.org.

[14] AMQP, DOI= http://www.amqp.org.

[15] Bernstein, D. Using XMPP as a transport in Intercloud Protocols, DOI= http://www.cloudstrategypartners.com/6.html.