Service Provisioning in Content-Centric Networking

Challenges, Opportunities, and Promising Directions

Contemporary IP-based Internet architecture increasingly is unable to meet the demands of current network usage patterns. Content-centric networking (CCN), as a clean-slate future network architecture, is different from existing IP networks and has some salient features, such as in-network caching, name-based routing, friendly mobility, and built-in security. This architecture has a profound impact on how Internet applications are provisioned. Here, from the perspective of upper-layer applications, the authors discuss challenges and opportunities regarding service provisioning in CCN. They also describe the Service Innovation Environment for Future Internet, their approach that addresses challenges while exploiting opportunities for the future of CCN.

Xiuquan Qiao and Junliang Chen
State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications

Wei Tan
IBM T.J. Watson Research Center

Schahram Dustdar
TU Wien, Austria

Content-centric networking (CCN)[1] — also known as named data networking[2] — is a new networking paradigm centered on content distribution rather than host-to-host connectivity. It aims to change the Internet from IP addresses to named data, to address the challenges faced by existing IP networks, such as scalable content distribution, location independence, mobility, and security. CCN decouples the content from the location and provides in-network caching, leading to reusable resources cached in routers closer to users. In CCN, content is named at the granularity of individual packets. CCN offers two distinct packet types in the network: an interest packet and content packet. Both packet types carry a name, which uniquely identifies a content segment. Each packet is routed and forwarded based on its name independently.

Specifically, each CCN router maintains three major components: forwarding information base (FIB), pending interest table (PIT), and content store (CS). When a consumer sends out an interest packet, which carries a name that identifies the desired data, the CCN router looks up its CS to find out whether it has a cached

 1089-7801/16/$33.00 © 2016 IEEE

copy of the data. If so, the router responds with the cached copy. Otherwise, it records the interface from which the interest packet comes into the PIT, and forwards the interest packet by FIB. The router stores the interest packet in its PIT along with the interface from which the packet has been received, until the expected data are received. When more than one interest packet for the same data arrive, only the first packet is forwarded upstream toward the data source, and other incoming interest packets are recorded in the existing PIT entry. When the interest packet eventually reaches a node with the desired data, the content packet, together with a signature by the producer's key, is returned by tracing back the footprints in the PIT left by the interest packet. Each data packet is a self-identifying and self-authenticating unit, which provides a built-in data security mechanism.

In recent years, CCN has attracted considerable attention in both academia and industry. However, the existing research work mainly focuses on the network layer. So far, little attention has been paid to the impact on existing Internet content providers with regard to service provisioning in such a network architecture. We argue that, upper-layer applications, rather than lower-layer network protocols, should be the fundamental driving force of Future Internet architectures. Currently, service provisioning in CCN is largely ignored. This will greatly hamper the wide adoption and commercial deployment of CCN. Compared with the existing IP networks, CCN has salient features, such as in-network caching, name-based routing, and content awareness. With the in-network caching and the interest packets' aggregation features, CCN reduces the content server's load, network traffic and content access latency. The consumer's mobility is addressed by employing the publish/subscribe communication model in CCN, which makes the provisioning of mobile Internet applications easier instead of using the existing mobile IP approach. In addition, CCN is built on the notion of content-based security. These new network features could lead to some service-provisioning challenges and service-innovation opportunities for both network operators and application providers.

Here, we discuss the challenges and innovation opportunities involved in service provisioning in CCN. Based on this analysis, we propose the Service Innovation Environment for Future Internet (SIEFI) architecture as an infrastructure and testbed for Future Internet application innovations and experiments.

Challenges in Service Provisioning

As a clean-slate Future Internet architecture, the CCN architecture is evolving quickly and facing multiple challenges, such as name-based routing lookup, an in-network caching policy, and mobility issues with content publishers.[3] Here, we mainly focus on the application-provisioning challenges faced by CCN from four aspects: application tools, application evolution, a business model, and application friendliness.

Lack of Rich Application Tools Specific to New Networks

Current application tools are mostly designed for existing IP networks and related application protocols. These existing application tools can't directly support CCN-based applications. Obviously, it will require tremendous efforts to rewrite or update current application tools running over IP to run over CCN. Therefore, the lack of application tools natively supporting the CCN system will greatly hamper CCN's wide deployment.

Taking Web applications as an example, CCN will have a great influence on Web architecture. These days, Web applications have become one of the fundamental Internet services. In an IP-based network, when a client needs to fetch a Web resource, first the network looks up the DNS server to translate the domain name to the IP address, and then the server sends the IP packets to routers, which are responsible for transferring the packets from the source address to the destination address. To enhance the system performance and reliability, the large-scale Web systems often use load balancing to distribute workloads across multiple webservers. To further provide end users with high availability and high performance, the content distribution network (CDN) provides content through servers deployed in multiple data centers across the Internet. Obviously, in an IP network, the provisioning of Web applications becomes increasingly complicated. However, in CCN, the provisioning of Web applications will be greatly simplified. The client can send interest packets with a content name directly to routers and fetch content packets from either the intermediate routers' cache or origin webserver. A large number of duplicate static content requests are processed by the network itself and only a small number of content requests will arrive at the server. We can see, then, that the existing complex DNS and CDN approaches for Web applications are all replaced by the native CCN system itself. Hence, the Web application provisioning in CCN becomes much easier.

However, Web applications deployed on CCN require a dedicated Web browser and webserver that can support CCN inherently. From the specific implementation, there are huge differences in terms of protocols and communication patterns. Specifically, the existing Web applications are built on HTTP/TCP/IP protocol stacks and it's easy for the client to fetch, update, and submit content to the server by general HTTP methods. However, in CCN, Web applications are built on an interest packet/content packet protocol stack. Communication in CCN is driven by the data consumer. All in all, the clean-slate CCN architecture requires the new application tools to adapt to it. Therefore, researching and developing new application tools dedicated to CCN is an enormous challenge in terms of the wide deployment of CCN. Currently, application researchers are working to develop a variety of application tools dedicated to CCN, such as Voice over CCN,[4] Chat, Filetransfer, Videostreaming, audio conference, P2P, as well as the Web browser[5] and webserver[6] developed by our team.

Limited Methodology for the Evolution of Existing Applications

Despite its advantages, realistically CCN won't replace the existing IP network immediately. Therefore, a smooth evolution — that is, maintaining compatibility with IP-based applications for an extended period, is not only desirable but necessary. One approach is to use a proxy or gateway to redirect the existing application traffic to the new network. This approach makes existing applications, webservers, and development tools unchanged, with an overhead of protocol conversion. For example, application researchers developed an HTTP/CCN proxy/gateway to import the real Web traffic into a CCN system.[7] An alternative is to update the existing applications, servers, and tools to be compatible with both the IP network and CCN, such as our previous work: a CCN-enabled Web browser[5] and webserver.[6]

Potential Mismatch with Existing Business Models

Ubiquitous in-network caching and interest packet aggregation are two important features of CCN. CCN routers can directly respond to the interest request using their embedded content stores and aggregate the same interest packets with the help of a PIT. By this means, requests to popular content likely won't reach the origin content source, but be served by intermediate routers. With this feature, CCN reduces the content server's load, network traffic, and content access latency. The disadvantage is that content providers will be unaware of the actual content usage throughout the network. In the current Internet economy, content providers rely on advertisements accompanying the content to generate revenue. CCN will present content providers with a challenge, in the sense that content created by a provider probably won't be served by it. This new service model is misaligned with existing business models, where advertising revenue is based on the content hit count, and might lead to content providers becoming concerned that their content could be illegally spread. Therefore, it's necessary to give consideration to content providers' interests and concerns, resolving this dilemma through new technologies and regulations, while also leveraging the advantages of CCN.

Because CCN uses a totally different data transmission primitive than the traditional socket-based primitive of a TCP/IP network, the provisioning of hits-based content services in CCN faces some challenges. On the one hand, not all content requests will reach the content server, so the counting of content use on the server side becomes difficult. On the other hand, it should be noted that CCN works at a named-data level, which means a large file is chunked into multiple small pieces and encoded into data packets. The upper application often needs to send multiple interest packets with different segment numbers to request the corresponding data packets. In this way, a user's visit might generate multiple interest packets in CCN routers. For a visit to the content, some interest packets are satisfied by intermediate routers and other interest packets are answered by the origin content server. In this case, the content server is hardly able to identify a single visit behavior for the content. Therefore, determining how to satisfy the realistic business needs of content providers is also a technical problem to be explored for real CCN deployment. To tackle this, we first tried exploring the hits-based content-provisioning mechanism in CCN.[8]

The Need for Efficiency When Supporting Diverse Application Types

Besides the pull-type static content distribution service, the existing IP-based Internet also possesses various dynamic interaction and push-type applications. For example, in Web 2.0, user-generated content, such as those in social network websites, is becoming increasingly popular. In this case, submitting personalized data to Web servers is a typical push service. Today's IP-based Internet uses a sender-controlled model that can easily support both pull- and push-type applications. However, CCN adopts a receiver-controlled way that works in a pull mode. The original CCN interest packet hardly considers the issue of personalized user data. In this paradigm, the conventional sender-controlled, push-type manner isn't efficient in CCN.

In essence, CCN is more suitable for consumer-driven, pull-type static content dissemination. When the client wants to submit some personalized content to the server side, or the server side wants to push some personalized content to the client side, the pull mode of CCN would act inefficiently. As an application-neutral network, CCN should efficiently support diverse application types. Therefore, how to efficiently support the push-type application mode is a problem to be addressed in CCN, and one which application researchers are currently exploring.[9] Fortunately, the recent CCN implementation (CCNx 1.0) has fixed this drawback in its protocol stack. In this new version, the CCN interest packet message supports the optional payload that can be used to carry a user's personalized data.

Application Innovation Opportunities

The existing IP network is used as a dumb pipeline, and the intelligence remains in end hosts at the network edge. While in CCN, in-network caching and name-based routing are two important features, and routers are therefore aware of the content transmitted. These new features will provide innovation opportunities for network operators.

In-Network Search with In-Network Caching

In-network caching in CCN brings a valuable content pool for network operators. Analyzing and using it can be not only interesting, but also rewarding.

One possible innovation for network operators is enabling in-network search. In existing IP networks, the content search services can only be implemented at the application layer. However, in CCN, network operators can index and search content in the network layer and in a cross-application manner. Meanwhile, the human-readable, hierarchical, content-naming scheme makes it possible to search by content name, such as *Titanic* or *Kung Fu Panda*. CCN also supports search functions. For example, when the user sends the interest packet with the content name "ccnx://baidu.com/weather/" and the value of the selector field "MaxSuffixComponents" is 2, the router might return the data packet with the name "ccnx://baidu.com/weather/Beijing/2015-04-01" in content caching. From this, we can see that the current CCN network only shares and reuses the same content from the same provider; it can't decouple the content from the provider. For example, the current CCN implementation considers "/baidu.com/movie/**Titanic**.mov" and "/youku.com/movie/**Titanic**.mov" as two different content pieces. But in fact, end users often only care about what the content is (that is, the content semantics); they don't care about who provides the content and from where. Because routers might store different content providers' contents, network operators are able to search from different providers for users. For example, when a user inputs the keyword *Titanic*, the routers can return "/baidu.com/movie/**Titanic**.mov" or "/youku.com/movie/**Titanic**.flv" from in-network caching. Then, the name semantics-based content search service ultimately decouples content, location, and providers.

Therefore, in the Future Internet architecture, network intelligence (built on name-based routing) will be greatly enhanced and the network could become a huge content search service cloud. Clients only need to present what content they want, regardless of where it is and who provides it, and the network will automatically complete a content search, along with transmission and dynamic resource scheduling. This new Internet architecture could potentially bring a new kind of Internet search service delivery model for network operators. However, network designers fear that this in-network search could result in high costs for additional traffic and router processing complexity. As a workaround, they still tend to implement the search function in the application layer. However, some application researchers think that in-network content search should be as a basic function of CCN. At present, there's no consensus on this issue. Therefore, whether the CCN routers should have this feature is still an open question.

Internet Traffic Analysis with Name-Based Routing

Existing IP routers can hardly detect the content of Internet packages. Although Deep Packet

Inspection (DPI) technology enables the existing network to become a content-aware network, its overhead cost is gigantic. Therefore, existing traffic analytics for content usage trends are happening on the application layer. In the meantime, because CCN employs name-based matching and routing, routers are aware of the content requested. Therefore, name-based routing makes it convenient for network operators to analyze the content access patterns. From the perspective of network operators, content request analysis is of great commercial value. For example, network operators can analyze content access trends, and provide competitive analysis, benchmarking, market research, or business development proposals. This business model can adopt the well-known Alexa model (www.alexa.com), but with more details regarding content.

When collecting content request logs, protecting privacy is also important. Unlike the conventional IP packet with the source address and destination address, the interest packet of CCN has no personal user identifier or end-host address information. Therefore, in the network layer, the CCN routers don't know who sends the request and only know which content segment is requested by which interface. Therefore, this approach only collects and analyzes the content access logs and has no user privacy information. Thus, this approach is safe for consumers.

In CCN, due to in-network caching and interest packet aggregation features, all routers can answer or aggregate the interest packet request. Hence, core routers might only receive some, but not all, of the interest packets. Only the edge routers can fully reflect the real content request amount. Therefore, the edge routers should be selected as the collection point of interest packets representing content visit behaviors. However, in real network deployment, a potential issue is that not all edge routers can be governed by the network operators, because some edge routers are owned by some individuals or enterprises, which might result in the inability of some network operators to document access behaviors. However, because most requests can be collected by network operators, these statistical results basically reflect the content access patterns. Therefore, they have no substantial influence on the analysis result of content usage in the network. Another risk is that the different ISPs might be unwilling to share their information with each other, but content providers can subscribe to their information from different network operators.

Personalized Web User Interaction Interface

Users as content consumers are only interested in the content itself, rather than its locations or providers. Ideally, the user interface should focus on content navigation, recommendations, and search. This requires a transformation from the existing URL-based model to a content name-based one. The user interface might screen the content's location and provider. The same content could have more than one location and provider, but that information is transparent to the users, and the network will be responsible for content location and search from the nearest content sources. In addition, because the network can easily sense trending topics, and a client (such as a Web browser) has access to information about the user's favorite sites, it's possible for network operators to provide a personalized user interface in the application layer according to the latest topics, integrating the user's preferences.

A Promising Direction

To facilitate Future Internet innovations, several future network innovation testbeds have been established, such as the Global Environment for Network Innovations (GENI; www.geni.net)[10] and Future Internet Research and Experimentation (FIRE; www.ict-fire.eu/home.html). Despite the existence of network testbeds, there's a lack of testbeds for upper-layer application research and innovation. When researchers need to conduct application experiments on new network architectures, they must download the related application tools and install them on their own machines. Thus, to conduct a large-scale performance testing, they must set up the scalable computer clusters' deployment environment.

This is a complicated task for most application researchers. Therefore, we argue that it's necessary to build something like Service Innovation Environment for Future Internet (SIEFI) on top of the existing future-network testbeds, to provide application tools and runtime engines to support service innovation in the Future Internet. For example, users can set up a CCN experiment network in the underlying Future Internet testbed. SIEFI will provide a new Web browser and webserver dedicated to the CCN network; that facilitates the researcher's efforts to conduct Web application experiments on the CCN experiment network. With its simple configuration, the CCN Web browser[5] and webserver[6] in SIEFI can communicate with the underlying CCN routers in the Future Internet testbed.

SIEFI

To this end, SIEFI must provide a series of service-development tools, service-execution engines, and rich network service resources to enable researchers to efficiently build applications oriented toward the Future Internet. The specific goals of SIEFI follow.

Explore Future Internet architectures to better support existing Internet services and applications. As we mentioned, the clean-slate Future Internet architectures have different running principles from the existing IP networks, which will result in the service-provisioning approach changing dramatically. Therefore, we must explore new service-provisioning principles for existing mainstream Internet services and applications in the Future Internet. In addition, the smooth-evolution approach of existing applications must be verified.

Explore opportunities for innovative services. The new network is designed to address the challenges faced by existing networks. There are some new native features, such as in-network caching, name-based routing, content awareness, mobility, and security. These new network features could offer a number of new opportunities for service innovation. SIEFI can provide a service innovation environment integrated seamlessly, with an underlying new network to foster the upper-layer applications. By lowering the technology threshold of service development and innovation, SIEFI can benefit from the power of collective intelligence once researchers, users, and small and medium enterprises explore its innovative services.

Blueprint of SIEFI's Architecture

To achieve the aforementioned objectives, from a technical perspective SIEFI should consist of the key components shown in Figure 1.

Service-development tools. The service development tools are mainly used to assist the user in efficiently developing applications. Because different application requirements have different development characteristics, to meet the diverse and personalized services innovation needs, SIEFI should provide a set of common service development tools in a suite — such as new Web, streaming, and mobile applications oriented to CCN, a workflow-development tool, and a service-testing tool. This will enable users to develop various network applications flexibly and efficiently. The service-development tools will be adopted in an open-management mode. Thus, the tool providers can always upload and update the relevant tool information.

Service-execution engines. The applications developed with different development tools need the appropriate back-end service execution engines to load and monitor the service's operation. Therefore, corresponding to different types of service-development tools, SIEFI should provide the appropriate service-execution engines to support the deployment of services, including current mainstream execution engines and new service-execution engines adapting to future networks.

Service resource library. The service resource library is an important service asset repository of SIEFI, aiming to share and reuse network service resources. Some common and domain services are specified by a standard service interface language such as Web Service Definition Language (WSDL) or Restful, covering telecommunications services, social networking services, audio and video encoding and decoding, and location services. With these existing service resources, users can efficiently generate their own applications with the aid of service-development tools. The continuous accumulation of these public service resources will lay the foundation for the rapid generation of subsequent services.

Service management. The service-management module provides full-lifecycle service-management functions, including service authentication, user management, service deployment, and resource management. In addition to reusing the existing services, SIEFI should provide a service migration ability, which enables users to migrate their existing applications to SIEFI. By providing users with a one-stop service portal, SIEFI lets users conveniently use the authorized services and resources provided by this facility.

System integration bus and cloud-based service-deployment platform. To support scalability and openness, SIEFI should adopt an open system architecture that supports the integration of various service-development tools and service-execution engines. In this way, any existing and new tools dedicated to future networks can

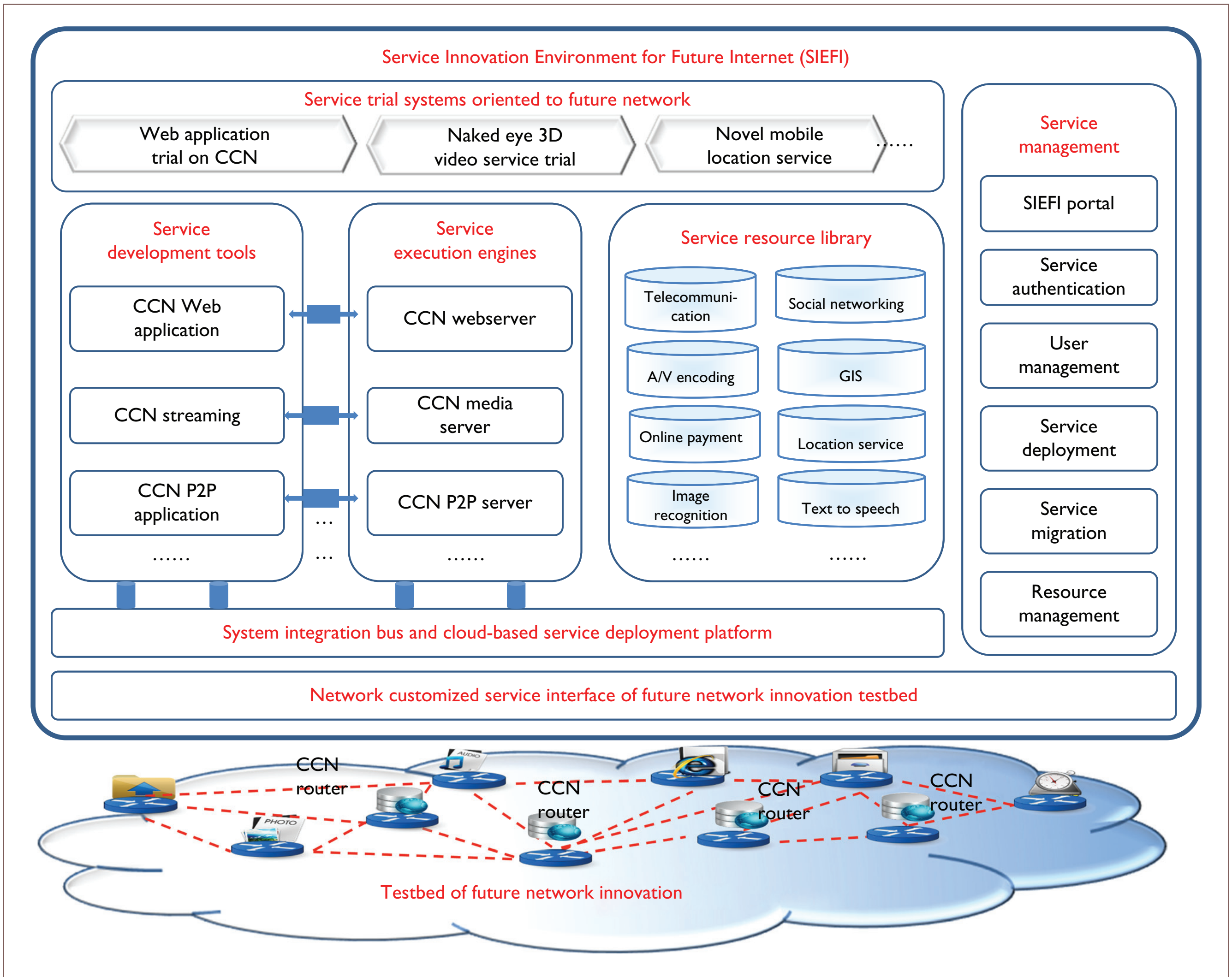

Figure 1. Architecture of the Service Innovation Environment for Future Internet (SIEFI). CCN stands for content-centric networking.

all seamlessly integrate with SIEFI. Also, SIEFI should be built as a highly reliable, scalable cloud service platform that lets users automatically deploy and manage the developed services,[11] and provide dynamic expansion and load-balancing capacity. Then application developers won't have to worry about the high cost of service operation and maintenance, and can focus purely on the application logic, reducing the service-provisioning cost for developers.

Customized network service interface for future network innovation testbeds. In general, the existing testbeds for future network innovation all employ programmable routers to build a flexible experimental environment. By separating the network control plane and data plane, users can flexibly customize the underlying network according to the different application requirements. For example, the upper-layer applications can build the virtual networks with different qualities of service. This flexible programmable mechanism removes the gap between the underlying network and the upper-layer applications. Therefore, to make the upper application more cohesive with the underlying network, SIEFI should provide the underlying customized network service interfaces to the upper-application developers.

The CCN paradigm presents upper-layer application provisioning with some challenges and opportunities. To facilitate application research activities with respect to the Future Internet, SIEFI bridges the gap between the upper-layer applications and the underlying Future Internet infrastructure. Currently, we're developing a prototype of SIEFI to demonstrate its advantages. We recognize, however, that a much larger and more systematic debate for such change is needed. If our work fosters deeper thought and offers a way to promote and organize existing efforts on these topics, we think it's meaningful.

Acknowledgments

This work was supported by the National Key Basic Research Program of China (973 Program) under grant 2012CB315802; National Natural Science Foundation of China under grants 61171102 and 61132001; Prospective Research on Future Networks of Jiangsu Future Networks Innovation Institute under grant BY2013095-4-01; Beijing Nova Program under grant 2008B50; and Beijing Higher Education Young Elite Teacher Project under grant YETP0478.

References

1. V. Jacobson et al., "Networking Named Content," *Comm. ACM*, vol. 55, no. 1, 2012, pp. 117–124.
2. L.X. Zhang et al., "Named Data Networking," *ACM Sigcomm Computer Comm. Rev.*, vol. 44, no. 3, 2014, pp. 66–73.
3. G. Xylomenos et al., "A Survey of Information-Centric Networking Research," *IEEE Comm. Surveys and Tutorials*, vol. 16, no. 2, 2013, pp.1024–1049.
4. V. Jacobson et al., "VoCCN: Voice-over Content-Centric Networks," *Proc. 2009 Workshop on Re-Architecting the Internet*, 2009, pp. 1–6.
5. X.Q. Qiao et al., "NDNBrowser: An Extended Web Browser for Named Data Networking," *J. Network and Computer Applications*, vol. 50, Apr. 2015, pp. 134–147.
6. X.Q. Qiao et al., "CCNxTomcat: An Extended Web Server for Content-Centric Networking," *Computer Networks*, vol. 75, Part A, Dec. 2014, pp. 276–296.
7. S. Wang et al., "On Adapting HTTP Protocol to Content Centric Networking," *Proc. 7th ACM Int'l Conf. Future Internet Technologies*, 2012, pp. 1–6.
8. X.Q. Qiao et al., "Exploring Hits-Based Content Provisioning Mechanism in Content-Centric Networking," *China Comm.*, vol. 11, no. 7, 2014, pp. 24–39.
9. K. Kim et al., "A Push-Enabling Scheme for Live Streaming System in Content-Centric Networking," *Proc. 2013 ACM CoNEXT Student Workshop*, 2013, pp. 49–52.
10. C. Elliott, "GENI: Opening up New Classes of Experiments in Global Networking," *IEEE Internet Computing*, vol. 14, no. 1, 2010, pp. 39–42.
11. Y. Wei, and M.B. Blake, "Service-Oriented Computing and Cloud Computing: Challenges and Opportunities," *IEEE Internet Computing*, vol. 14, no. 6, 2010, pp 72–76.

Xiuquan Qiao is a full professor at Beijing University of Posts and Telecommunications (BUPT). His research interests include application innovations in Future Internet. Qiao has a PhD in computer communications. Contact him at qiaoxq@bupt.edu.cn.

Junliang Chen is a professor at Beijing University of Posts and Telecommunications (BUPT). His research interests focus on services computing. He's a member of both the Chinese Academy of Sciences and the Chinese Academy of Engineering. Currently, he's the Editor-in-Chief of *China Communications* magazine. Contact him at chjl@bupt.edu.cn.

Wei Tan is a research staff member at the IBM T.J. Watson Research Center. His research interests include Big Data and cloud computing. Tan has a PhD in control science and engineering from Tsinghua University. Contact him at wtan@us.ibm.com.

Schahram Dustdar is a full professor of computer science (informatics) and he heads the Distributed Systems Group at TU Wien, Austria. His work focuses on Internet technologies. Dustdar is an IEEE Fellow, a member of the Academia Europaea, an ACM Distinguished Scientist, and recipient of the IBM Faculty Award 2012. Contact him at dustdar@dsg.tuwien.ac.at.