Prototype of an Architecture for Object Resolution Services in Information-Centric Environment

Sripriya Srikant Adhatarao*, Jiachen Chen*, Mayutan Arumaithurai*,
Xiaoming Fu* and K.K. Ramakrishnan†
*University of Göttingen, Germany. Email:{adhatarao,jiachen,arumaithurai,fu}@cs.uni-goettingen.de
†University of California, Riverside, USA. Email: kk@cs.ucr.edu

1. INTRODUCTION

Information-Centric Networking (ICN) is a new paradigm where information is exchanged using the *Names* of the content. Recent ICN proposals like Named Data Networking (NDN [1]) and Content Oriented Publish/Subscribe System (COPSS [2]) use hierarchically structured Names and Content Descriptors (CDs) as the identity of the content. They assume the existence of some mechanisms that assist users in retrieving the "ICN-names/CDs" for the information they seek. However, these proposals lack a description of how users can obtain these "ICN-names/CDs".

In a recent work [3], we proposed an architecture for Object Resolution services in Information-Centric Environment (ORICE) that tries to mitigate this gap and thereby enhances the usability of ICN. It can be a framework for building object resolution systems in ICN. ORICE allows the object resolution systems to provide the service which may involve complex logic and large data access at the application layer that enables ICN to handle the communication in a simple and yet efficient manner. With ORICE, multiple object resolution systems can be deployed as per the need in the application layer. These systems can even communicate with each other and provide better services to users.

With the ICN as the underlying networking architecture, object resolution systems can be benefited with diversity in the recommendations along with service scalability for distribution of services and balancing the load in the network. Users can also get retrieval/dissemination efficiency thanks to the name-based routing and in-network caches along with data integrity.

In this demo, we present a prototype of ORICE, using ICN as the underlying network and demonstrate the feasibility of ORICE in fulfilling the necessity for object resolution services in ICN. We implement multiple object resolution systems using the architectural primitives proposed in ORICE and show how users can translate keywords to the names/CDs they might be interested in before retrieving the data from the underlying network.

ICN'15, September 30 - October 02, 2015, San Francisco, CA, USA
ACM 978-1-4503-3855-4/15/09.
http://dx.doi.org/10.1145/2810156.2812608.

2. PROTOTYPE OF ORICE

In this section, we describe the prototype implemented using ORICE. In particular, we demonstrate how clients can search for names/CDs in ICN (for query/response and publish/subscribe). The prototype is built on top of CCNx 0.8.0 and COPSS. We use Fig. 1 to demonstrate the working of our prototype. Here we have two object resolution services (ORS_1 and ORS_2, having 1 server each), 1 certification server (CS) and 2 users (U_1 and U_2) attached to a network consisting of 6 routers (R_1–R_6). We also implemented the broker design in COPSS ($Broker$ in Fig. 1) to provide support for asynchronous data dissemination. Brokers subscribe to their responsible CDs and receive publications that can be requested by offline users. The certification server, resolution server and broker system register their respective prefixes (*i.e.*, `/CertPrefix`, `/ORSx` and `/Broker`) to ensure that CCNx can route the Interests to the intended server based on the prefixes. The resolution servers subscribe to the management channel (`/ManageChannel`) on which they receive updates in the name space from certification server. We demonstrate the system with the following scenarios.

2.1 Scenario [Post]

Data providers have the responsibility for associating appropriate names/CDs to the content they post in the network. ORICE allows them to seek names/CDs from the object resolution services.

As shown in Fig. 1a, when a user U_1 wants to publish a piece of data, he would type the title and content of the message in the graphical user interface (GUI) as shown in Fig. 1b. Instead of typing the name and CDs himself, the user can get suggestions from an object resolution service. We provide 2 basic object resolution services and the user can set the preference by setting the parameter "default search engine prefix". The application would send a request using the format `/ORSx/Search/Message` and the request will be routed to the corresponding object resolution service and receive the suggestions (the green arrow in Fig. 1a shows a request goes to ORS_2).

U_1 can choose any name/CD for the message before publication. If he is not satisfied with the existing identities, he can request to add a new identity by sending an Interest with name=`/CertPrefix/ID` and content="add". This packet will be forwarded to the certification service CS and U_1 will get the response indicating if the request is approved (flow not shown in Fig. 1a). Upon approval, CS will notify the changes to the name space via the management channel (with CD=`/ManageChannel/CD`) to the respective resolution servers (the red arrows in Fig. 1a).

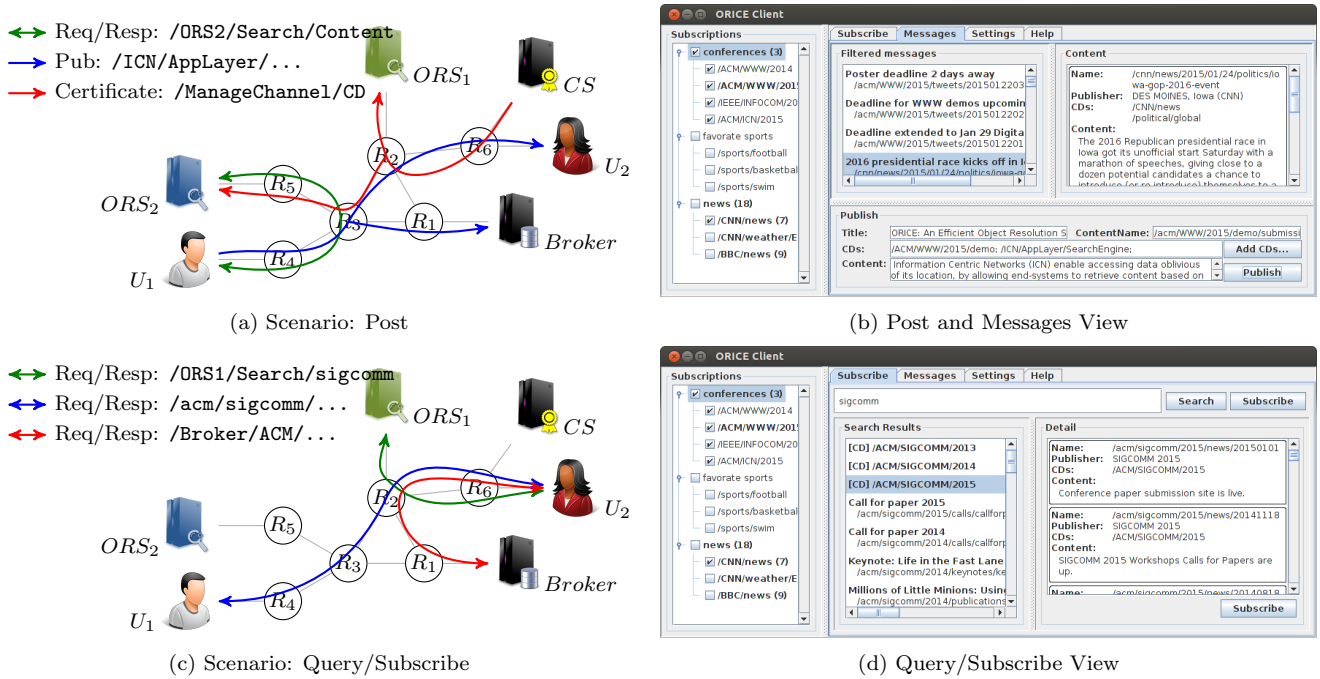

Figure 1: Example scenario and GUI of ORICE prototype.

When U_1 clicks the "Publish" button, the application will start to serve the message and also multicast the message to the users who have subscribed to the related CDs. In the example, U_2 is subscribing to the message and her GUI will place a notification on receiving the multicast. In the demo, we use a broker (described in [2]) that listens to all the CDs as a backup server for the application. The blue arrows in Fig. 1a shows the flow of the multicast and Fig. 1b shows that the received messages are grouped into different topics based on the CDs. The user has 3 new messages in topic "conferences" and 7 new messages in CD `/CNN/news`. She can browse through the topics, CDs and also read the details of the messages in the GUI.

2.2 Scenario [Query/Subscribe]

If a user is interested in a topic, in the network (s)he can either issue a query (for past data) or perform a subscription (for future data). To simplify the procedure, we unify the two processes in our demo. The user can simply type the key words (s)he is interested in as shown in Fig. 1d.

When the "Search" button is clicked, the application sends a request using the default search engine prefix (`/ORS1/Search/sigcomm` in the example, the green arrow in Fig. 1c). ORS_1 responds with a list of candidate names and CDs and the application lists the results in the middle column in Fig. 1d.

If a user is interested in a piece of data (*e.g.*, the 4th item in the list), (s)he can click on the item and the application would send a request using the ContentName in the entry. In the example, U_2's application requests for the data `/acm/.../callforpapers.pdf` and the packet is forwarded in the network following the blue arrow. This kind of requests can fully exploit the benefits provided by ICN. *E.g.*, if R_3 has a cache or *Broker* already has the data, the request will be redirected before it goes all the way to U_1.

When a CD entry (starting with "[CD]") is clicked, the application would try to request the broker for the most recent messages. It would send a request with name `/Broker/CD` and put "latest" as the selector (the red arrow in Fig. 1c) and the broker would reply with a list of data published recently as is shown in the GUI. These messages help the user to decide if the CD is the one that (s)he is interested in. When the user decides to subscribe to the CD, (s)he can click the "Subscribe" button in the bottom right corner and the application will subscribe to the CD (flow not shown in Fig. 1c).

When the clients come back online, the application can request the broker system using the name `/Broker/CD` and the selector to filter out all the received messages. *Broker* responds with messages accordingly.

For demonstration, we have set up the test bed as shown in Fig. 1a and Fig. 1c. With user interfaces as shown in Fig. 1b and Fig. 1d, we will show two object resolution systems implemented using ORICE framework each running on a different computer. With GUIs like Fig. 1b and Fig. 1d, we show users interacting with the object resolution systems. We will deploy the ICN platform in our lab test bed and the front-end applications will access the platform through SSH tunnels.

Acknowledgment

This research was partly funded from the EU-JAPAN FP7-NICT GreenICN project, the Volkswagen Foundation Project "Simulation Science Center" and the US National Science Funding under Grant No.CNS-1455815.

3. REFERENCES

[1] V. Jacobson *et al.*, "Networking Named Content," in *CoNEXT*, 2009.

[2] J. Chen *et al.*, "COPSS: An Efficient Content Oriented Publish/Subscribe System," in *ANCS*, 2011.

[3] S. S. Adhatarao *et al.*, "ORICE: an Architecture for Object Resolution Services in Information-Centric Environment," in *LANMAN*, 2015.