Distributed Service Recovery for Real-Time Pub/Sub Services on Disaster Management

Chi-Sheng SHIH, and Hsin-Yi CHEN
cshih@csie.ntu.edu.tw
Embedded Systems and Wireless Networking Lab
Graduate Institute of Networking and Multimedia
Department of Computer Science and Information Engineering
National Taiwan University

***Abstract*—How to exchange information between parties in a mega-scale disaster management system is one of the fundamental challenges to support timely and efficient disaster response and relief. Specifically, the timeliness, scalability, and availability are three desirable features for information exchange. We call the framework to support information exchange with the three features an Open Information Gateway, OIGY in short. In this paper, we present the challenges of information gateway, and the design of the communication protocols and the fundamental components and distributed service recovery algorithms to support the aforementioned features. The algorithm aims at recovering the real-time publish and subscription services without the global information of network topology, available service brokers, and publish/subscription databases. A centralized algorithm is also presented for small scale networks.**

I. INTRODUCTION

Successful disaster response requires the collaboration from many parties including telecommunication service providers, web service providers, general public, rescue agencies, and rescue coordinators. When disaster occurs, how to exchange information among victims, rescuers, and decision makers is one of the most critical challenges. The goal of this work is to design and prototype a distributed information gateway to enhance responsiveness and availability of information exchange for disaster response.

In the last few decades, many attempts aimed at developing special communication devices and reserving specific communication channels for disaster rescue. Examples include satellite phones[1], IP-based 911[2] and rescue radio[3]. However, the applicability of these new technologies was founded limited. Take satellite phones as an example. Satellite phones provide location-free communications no matter whether the users are located in mountainous area, metropolitan, or on the sea. It is extremely effective for rescuers in mountainous area and sailors on board. Due to its high deployment cost, it is not possible to put a satellite phone in every emergency kit. In addition, it requires regular maintenance to assure its functionality. In the mountainous area of Taiwan, every village is equipped with one satellite phone for emergency use. However, due to poor maintenance, the batteries were mostly dead and many of the phones did not function at all during 2009 Morakot typhoon [4].

The experience in last several disaster rescue efforts show that how to effectively make use of all available communication devices and services is the key to a successful rescue effort. During Haiti earthquake [5], the victims trapped in the damaged buildings sent text messages via their cell phones, which allow the rescuers to locate them in the left-behind area and save more than 60 victims. During 2009 Morakot flooding in Taiwan [4], the destroyed communication infrastructure prevented the victims from contacting their family members and rescue agencies. However, their family members posted on plurk and twitter that the victims were not reachable, and marked their possible locations on maps.google.com. The information were broadcast by phones and social network web services. Consequently, the rescue team was able to locate the victims, and provided food and water supply to the victims. This information exchange model was proved to be effective for disaster response. However, such a successful rescue requires both effective coordination and timely intelligent information, which were conducted by experienced rescuers and/or crowd sources.

We call an information exchange framework that are designing for collecting, fusing, and distributing information during disaster management the *information system for disaster management*, and refer it to *ISDM* for short. An ISDM is capable of making diverse, multi-domain data and information generated by independently developed intelligent things and from people less fragmented and more trustworthy, delivering the information to independently developed disaster management applications and services with high availability and on a timely basis, and supporting different usages of the information for disaster preparedness, response and relief purposes and for research and planning in disaster reduction. The system can also adapt to needs, evolve and grow in capabilities with scientific and technological advances and can readily accommodate new information sources, applications and services as needed in response to unforeseen crisis situations.

An effective ISDM replies on many ICT (Information and Communication Technologies) components such as data repositories, fusion of symbiotic information, and information exchange.

In this paper, we are interested in information exchange services in ISDM. To support effective disaster response, the system must take into account the timeliness and run-time service composition for information exchange. In a disaster area, the physical and network infrastructure may be severely damaged or simply unavailable. On the other hand, before issuing rescue operations in response to urgent disastrous situations, acquiring current and accurate data will be the most critical operation. In general, data are collected from a

collection of pre-installed or quickly-deployed sensor devices, monitoring stations, satellite images, as well as civilian witness reports. Each of the data sources has its individual characteristics of physical properties (e.g. proximity of observation location), temporal properties (e.g. how often data are reported), numerical properties (e.g. sensitivity capability), and even rational properties (e.g. observations under human emotional stress). An ISDM must be able to select and integrate multiple data sources into a coherent information service. Hence, how to discover and compose information service in an efficient manner is a major challenge for ISDM.

It is evident that the research community and industry should continue their attempts to develop new sensing technology and affordable communication devices for disaster rescue. In the meantime, how to assure that existing and widely available (tele-)communication devices can be federated into the disaster management systems in harmony is critical to rescue efforts. Hence, this work is aimed at how to recover the communication network to assist victims and rescuers to communicate with the others in the disaster management system, how to orchestrate the information to reach their destinations, and how to acquire the information in timely manner. In short, the work is aimed at develop *Open Information Gateway*, called *OIGY* for short, which is an distributed middleware to support information exchange during disaster response.

In this paper, we are interested on real-time information exchange in ISDM. In particular, we focus on how to recover the services when parts of the real-time information publish and subscription malfunction. The remaining of this paper illustrates the preliminary study and design of Open Information Gateway. In Section 2, we illustrate the desired features of Open Information Gateway, its challenges, and related works. Section 3 illustrates centralized and distributed service recovery algorithms. Section 4 discusses the work to be completed in the near future.

II. Related Works and System Architecture

The information exchange services for disaster management has three desirable features: *scalability*, *timeliness*, and *availability*. In the following, we present the significance and challenges of these three desirable features. We will also discuss the latest development on these three desirable features.

Many of modern information exchange services are (either *logically* or *physically*) centralized to reduce maintenance efforts. Client-server and web-based services are two typical examples of centralized services. This type of information exchange services have unified view of data and are easy to maintain; it also provides a single service point and provides better usability for the users. There are no doubts that this service model has its own merits. However, this service model highly depends on a reliable and robust network infrastructure so that the users can reach the service point. During a disaster, it is difficult, if not impossible, to have reliable and robust network connection. In addition, the type of services does not scale up. Although most of modern information exchange services make use of distributed architecture to share the load, none of them are designed to be scaled up in a short time window and to provide service for bursty requests.

Timeliness means that the information should be delivered within a given time interval. Most, if not all, of messages for disaster management are either time sensitive or time critical. Hence, timeliness is a critical requirement for information exchange in disaster management. Although the granularity of timeliness requirements for disaster management is much greater than that for safety critical systems, the outcome of late information exchange are also vital.

Availability refers to the availability of communication network, which is the most critical feature for ISDM. In this work, we also aim on recovering the communication network to connect the remote sensors, victims, and on-site rescuers with support teams. The rescue efforts in the last few disaster worldwide show that we should not rely on single type of communication network and have to integrate heterogeneous communication network to a coherent communication network. For instance, rescue personnel, mobile users, or tracked objects might move around the monitoring area. Long-range communication might be destroyed by the disaster. Hence, we need a flexible mobile infrastructure to cope with such uncertainty and temporary failures.

To achieve the above features, information exchange services must be designed as distributed information services. A distributed information exchange model allows ISDM to provide information exchange service when only parts of communication infrastructure is available. In addition, it also eliminates the bursty requests sent to a central portal. Publish/-subscription model is proved to be effective and efficient for systems that need repetitive, time sensitive data/message distribution. A publish/subscription system is comprised of information producers who publish and information consumers who subscribe to information. The key benefit of publish/subscribe for distributed event-based processing is the natural decoupling of publishing and subscribing clients. This decoupling can enable the design of large, distributed, loosely coupled systems that interoperate through simple publish and subscribe-style operations. Distributed publish/subscription systems are well-suited to handle large numbers of events occurred in large area networks. Distributed publish/subscription services do not rely on central publish and subscription services. The subscription requests are processed by pub/sub brokers located on different network locations, rather than centralized publication server. Consequently, distributed publish/subscription services can meet the scalability requirement of information gateway. PADRES [6], [7] and Web Solutions Platform Event System [8] are examples of distributed publish and subscribe services. However, modern distributed publish/subscribe services aim on distributing the information exchange services in the network and still rely on reliable network to provide a static network of pub/sub information brokers. When some of the brokers are either disconnected from the network or out of service, part of the services become unavailable. While managing disaster, the network is neither reliable nor

robust. Hence, the network of pub/sub information brokers must be dynamically formed. In addition, modern distributed publish/subscribe services do not balance or share the publish and subscribe requests on the broker. While managing disaster, the network can be congested or disconnected. Without managing the loads on the brokers, the services can have poor performance.

Real-Time publish/subscribe services extend publish/subscribe services to deliver messages in a given time interval. Real-time publish/subscribe well suits for the system with static network environment. One example is to use AMQP for stock exchange in Wall Street. Deng etc. [9] evaluated several real-time publish/subscribe service in QoS-enabled component middleware. This work empirically evaluates the performance of a container-based design and compare it with mature object-oriented real-time publish/subscribe implementations. One issue in most existing architecture that support publish/subscription is their limited support for the expression and enforcement of Quality of Service parameters (such as required latency, reliability, etc.). This is a significant shortcoming for existing architecture, which was addressed in this work. However, in the disaster management, the communication infrastructure changes over time. Existing systems do not address the change of communication infrastructure and will be address in our work.

Advanced Message Queue Protocol (AMQP) [10] originated in the financial services industry in 2006. AMQP is an open standard for message-oriented middleware (MOM) communication. AMQP grew out of the need for interaction between MOM systems both within, and between, corporate enterprises. Due to the proliferation of proprietary, closed-standard, messaging systems such integration is considered challenging. As such, the primary goal of AMQP is to enable better interoperability between MOM implementations. Since AMQPs inception, several, open-source, messaging software distributions have emerged. The Apache Qpid AMQP distribution includes a Broker federation option that can decentralize the architecture and share the workload among a group of Brokers linked with each other.

pubsubhubbub[11] is a distributed publish/subscribe communication over the Internet. The protocol resembles to existing RSS/Atom services, which allow the client to subscribe message services, and use the information polling protocol to retrieve updated messages. This protocol avoid the needs of periodically polling the feed server at certain arbitrary interval and reduces the network traffic during disaster response. Although this service allow the subscribers to receive pushed information, it still suffers from transient service failures and does not recover the services during disaster.

Content-centric Network (also content-based networking, data-oriented networking[12] or named data networking[13]) is an overlay network to allow the users to focus on the data they need, rather than having to specify a specific, physical location of a data sources. Hence, the connection is built with the name of the content, rather than the location of the content. Content-centric networking comes with potential for a wide range of benefits such as content caching to reduce congestion and improve delivery speed, simpler configuration of network devices, and building security into the network at the data level. Recent researches show that such an overlay network is feasible for certain network activities such as real-time multimedia applications.

III. System Architecture of Open Information Gateway and Real-Time PubSub

To support aforementioned desired features of information exchange services, we proposed to design and prototype an information exchange service framework, named *Open Information Gateway* (OIGY), to support distributed timely information exchange over heterogeneous networks. In this section, we present the methodology and advantages of each component in the system and how they interact with each other.

All the OIGY services will be executed on the edge devices in the network, including computationally weak devices such as cell phones and mud sliding sensors, and computationally powerful devices such as weather forecast service on cloud servers and data repository server in data center. Figure 1 illustrates the deployment of OIGY and software architecture of OIGY. As discussed earlier, numbers of individuals, news agencies, government agencies, and rescue agencies form the disaster management system. Each of them can be the providers and consumers for the information. For instance, data center in government agency can subscribe information from sensors in disaster area and publishes the fused/verified data to crowd sources. Victims in disaster area can publish their messages to the others and can subscribe information from their family members and government agencies. To achieve scalability, OIGY will be deployed as a middleware software component at each agency. For victims, an OIGY widget can be installed on their cell phones; for weather forecast agency, an OIGY service can be installed on their computationally power server. In addition, OIGY can also be distributed installed on Point-of-Service devices, which are located on supermarkets and convenient stores to publish and subscribe information. To deploy OIGY, minimal amount of software repository servers will be deployed to provide the middleware components for edge devices. However, these servers are not designed to provide any run-time service during disaster rescue.

Open Information Gateway (OIGY) consists of two major components: one is the distributed Truthful Real-time Information Publishing and Subscribing (TRIPS), and the other one is Heterogeneous And Plug-n-PlaY networks (HAPPY). The objective of HAPPY is to interconnect all network-capable devices using all kinds of possible manners, which may be comprised of heterogeneous network access technologies (e.g., WiFi, Bluetooth, Professional Mobile Radio (PMR), and 3G/GPRS) using different approaches (e.g., Infrastructure-based networks, wireless mesh networks, mobile ad hoc networks, and opportunistic networks). The objective of TRIPS is to support distributed real-time publish and subscription (P/S)

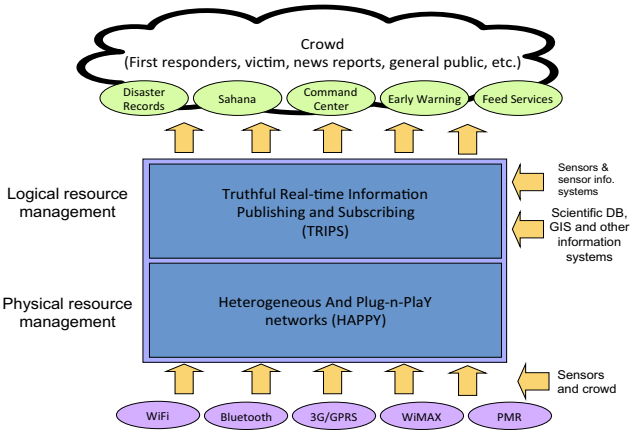

Fig. 1. System Architecture of Open Information Gateway (OIGY)

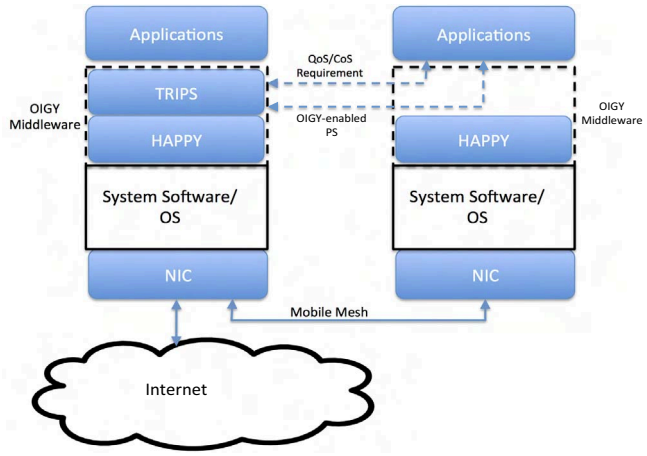

Fig. 2. Distributed Information Exchange in OIGY

services. Note that a device/service in the system is not limited to be either information publishers or subscribers. In fact, only few device/service can only be information publishers. They are the sensor devices to detect mudslide, and the sensors to measure rainfall and the water level in rivers. Most of the other services such as weather forecast services and GIS system will subscribe information from sensors, GIS databases, and other information sources and publish their information to the disaster management system and general public. While requesting for P/S service, the application specifies its the Quality of Service or Class of Service of its P/S service including timeliness, description, resource requirements, etc. TRIPS will register and announce the P/S service. Hence, one capability of TRIPS is to manage the declarations, automatically establish connections between publishers and subscribers for a matching topics and dynamically detect new status in the system. When the subscribed message arrives or is sent, TRIPS delivers the message to its subscribers. To address timeliness, OIGY will support distributed real-time message publish and subscription over large scale network.

Figure 2 illustrates the architecture of dynamic service composition and distributed information exchange. Compared to traditional publication and subscription services, the users in disaster management may not have the luxury and chance to the proper information sources. During the disaster, the volunteer tend to help to provide information in different formats. To take into account the dynamics and heterogeneity of the information services, TRIPS will take advantage of the SOA architecture to compose coherent information service. Based on our previous work on QoS management framework designed for SOA, we may treat each data source as a deployed service in SOA and map the data integration procedure as a service. With the dynamic service composition, TRIPS will be able to discover the publish/subscribe service when only parts of the communication network is available.

IV. Service Recovery for Distributed Real-Time PubSub Services

QPID is designed for exchanging messages for business process and assumes that the underlying IP network is static. Hence, there is no need to change the network topology of brokers during runtime and the message routing is completely handled by IP networks. To support disaster response, we modify the configuration process for QPID so as to support dynamic broker routing. Hence, in this paper, we use QPID as the base to illustrate the centralized service recovery.

A. Service Recovery with Landmark node

In the network, a *service node* refers to a computation process which executes one of the QPID task. Broker, exchange, client are few examples. On one physical computation node, one may deploy several QPID service processes. Hence, several service nodes are co-located on same physical process and communicate with each other via IP-based protocol. In the centralized service recovery framework, the broker networks are divided into several disjointed landmark subnets. Figure 3 illustrates the system architecture of the network. Each landmark subnet, represented by bold circle, has one *landmark node*, at least one *backbone* node, and several broker nodes. *Landmark node*, marked as shaded node in the figure, is a service node that is reachable from all the other broker nodes in the subnet and is responsible for monitoring and recovering the service in case of node failure. During runtime, the landmark monitors the messages sent by each broker node so as to monitor the liveness of each node. The monitoring process between landmark nodes and other broker nodes will be described later in the section. *Backbone node* is a service node which is responsible for transmitting messages between subnets. Depending on the topology of the network, there may be more than one backbone node in one subnet. In practice, we may deploy more than one service node on one physical node for the sake of simplicity. For instance, both of landmark

node and backbone node are deployed on a reliable computing node in the system to support reliable message delivery.

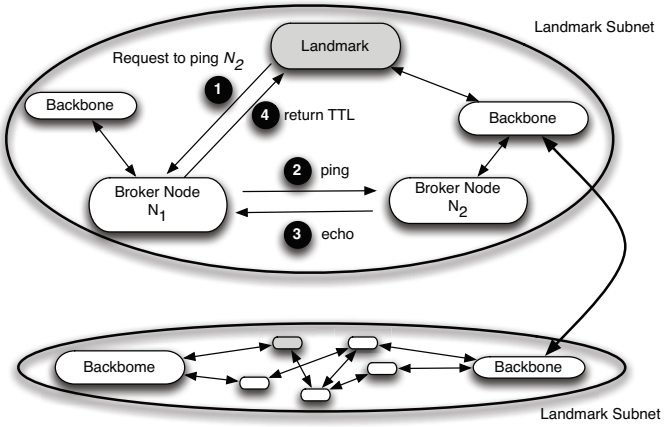

Fig. 3. System architecture and procedure for Ping daemon to collect route distance

Landmark node maintains the distance vector for all the broker nodes in the subnet. The vector keeps track of route distance between every two connected broker nodes and the distances are represented by the number of network hubs between two nodes. To obtain route distance between every pair of connected broker nodes, each computing node in the subnet deploys a ping daemon to collect route distance between itself and its connected nodes. To avoid flooding requests, route distance queries are initiated by landmark node. When landmark node sends a request to update route distance vector table, four operations are executed and are shown in Figure 3. Suppose that the landmark would like to update the route distance between broker node N_1 and N_2. It sends a request to node N_1 to collect route distance between N_1 and N_2. The ping daemon on node N_1 receives the requet and executes the ping utility to obtain the route distance between N_1 and N_2. When node N_2 returns the ping request to node N_1, node N_1 computes the route distance between these two nodes and replies to landmark node. If the network is not symmetric, landmark node will send the request to both broker node N_1 and N_2.

Landmark node listens to the events sent by each broker node. This operation does not consume too much resources because the landmark node listens to the messages broadcasted on the IP network and there is no additional message on the network. In addition, the landmark node stores the attributes of all the objects (entities) in the landmark subnet. When the landmark receives *broker disconnection event* from a broker node, which denotes a node failure, it starts the recovery process. First of all, landmark node selects another broker node as the alternative candidate to recover the failed service. The alternative candidate is selected based on the distance vector maintained by landmark node during the run-time. On the alternative candidate, landmark node creates new entities with the same attributes as the failed one and new links and routes. Last, it reconfigures the routing configuration on related broker nodes.

This mechanism uses a centralized approach to recover the failed services and is only suitable for a small landmark subnet. We will use this mechanism as the base to compare its performance with distributed service recovery mechanism.

B. Distributed Service Recovery

To enable distributed service recovery, the landmark node in each landmark subnet is replaced by numbers of *proposer node*s in the network. A proposer node is a broker node that monitors a set of broker nodes, which is defined by a monitor scope. A *N-proposer monitor scope* of proposer P_i is a set of broker nodes each of which is monitored by at most N proposers including proposer P_i. Figure 4 shows an example. In Figure 4, the shaded circles represent proposer nodes and the other nodes represent broker nodes. The example shows a 2-proposer monitor scope for proposer P_1, in which each broker node is monitored by at most two proposers. For instance, node P_4 is monitored by proposer P_1 and P_3 and node B_3 is monitored by proposer P_1 and P_2. In the meantime, node B_1 is only monitored by proposer P_1.

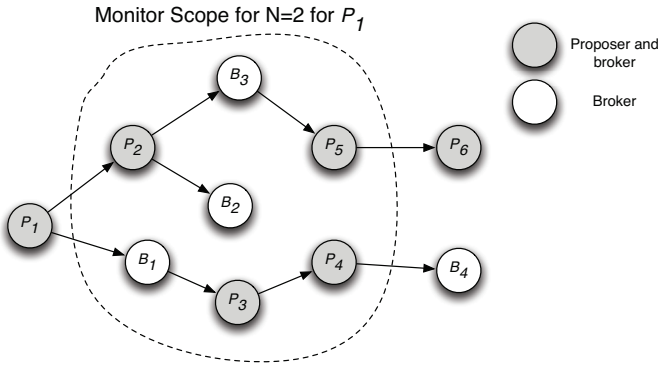

Fig. 4. Monitor Scope of Distributed Service Recovery

When detecting a failure, the proposer starts a session to find an alternative candidate to recover the service. Note that one broker node cloud be monitored by more than one proposer. Hence, these proposers may simultaneously detect the failure and starts the recovery session simultaneously. The proposer first finds an alternative candidate based on the distance vector table and forms a recovery proposal. The proposer sends the proposal to the arbitrator node, which is the immediate predecessor of the failed node, and waits for reply. If the proposal is accepted, the proposer starts to recover the service. Otherwise, it ends the recover sessions. If the arbitrator fails to response before timeout, the proposer starts to recover the arbitrator node.

When receiving a proposal, the broker node starts a recovery session as an arbitrator. If the proposal is expired or invalid (not repairing its children), it replies current network topology and ends this session. If the proposal is valid, it waits for the other $N-1$ proposals. When N proposals are collected

or timeout, it selects the optimal proposal, informs all the proposers, and ends the session.

The algorithm can recover the failed services when there are at most N concurrent nodes failure.

V. Summary

ISDM plays a critical role in disaster management. We proposed to design and prototype an open information gateway (OIGY) so as to delivery timely information in a distributed manner over heterogeneous networks. In this paper, we present the architecture for open information gateway and service recovery algorithms for distributed real-time publish subscription services. Centralized and distributed algorithms are presented and compared. The two algorithms will be implemented in QPID and evaluate their performance. The performance evaluation results will provide a guideline to deploy centralized or distributed recovery algorithms in practice.

Acknowledgements

This work was also supported by National Science Council under Grants NSC 99-2221-E-002-177-MY3, and by Academia Sinica under Grants AS-101-TP2-A01.

References

[1] W.-T. Chiu, J. Arnold, Y.-T. Shih, K.-H. Hsiung, H.-Y. Chi, C.-H. Chiu, W.-C. Tsai, and W. C. Huang, "A Survey of International Urban Search-and-rescue Teams following the Ji Ji Earthquake," Disasters, vol. 26, no. 1, pp. 85–94, Mar. 2002. [Online]. Available: http://doi.wiley.com/10.1111/1467-7717.00193

[2] U. S. D. of Transportation, "Next generation 9-1-1," last accessed at January 2012. [Online]. Available: http://www.its.dot.gov/ng911/

[3] U. S. C. Guard, "Rescue 21," last accessed at January 2012. [Online]. Available: http://www.uscg.mil/acquisition/rescue21/project.asp

[4] H. Chanson, "The Impact of Typhoon Morakot on the Southern Taiwan Coast," Shore & Beach, vol. 78, no. 2, pp. 33–37, Sep. 2011. [Online]. Available: http://espace.library.uq.edu.au/view/UQ:205351

[5] N. news services, "Rescue crews pull 2 more from haitian market," last accessed on January 17th, 2010. [Online]. Available: http://www.msnbc.msn.com/id/34829978/

[6] G. Li, H.-A. Jacobsen, and G. Alonso, "Composite Subscriptions in Content-Based Publish/Subscribe Systems," Lecture Notes in Computer Science, vol. 3790, pp. 249–269, 2005. [Online]. Available: http://www.springerlink.com/content/p3056525172037n6/

[7] G. Li, V. Muthusamy, and H.-A. Jacobsen, "A distributed service-oriented architecture for business process execution," ACM Transactions on the Web, vol. 4, no. 1, pp. 1–33, Jan. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1658373.1658375

[8] Keith Hamilton, "Distributed Pub Sub Event System," 2011. [Online]. Available: http://pubsub.codeplex.com/

[9] G. Deng, M. Xiong, A. Gokhale, and G. Edwards, "Evaluating Real-Time Publish/Subscribe Service Integration Approaches in QoS-Enabled Component Middleware," ISORC 07 Proceedings of the 10th IEEE International Symposium on Objectoriented Realtime Distributed Computing, pp. 222–227, 2007. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4208847

[10] S. Vinoski, "Advanced message queuing protocol," Internet Computing, IEEE, vol. 10, no. 6, pp. 87 –89, nov.-dec. 2006.

[11] Brad Fitzpatrick, Brett Slatkin, and Martin Atkins, "Pubsubhubbub core 0.3 working draft," , Tech. Rep., 2010. [Online]. Available: http://pubsubhubbub.googlecode. com/svn/trunk/pubsubhubbub-core-0.3.html

[12] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," in Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications, ser. SIGCOMM '07. New York, NY, USA: ACM, 2007, pp. 181–192. [Online]. Available: http://doi.acm.org/10.1145/1282380.1282402

[13] L. Zhang, D. Estrin, and J. Burke, "Named data networking (ndn) project," PARC Technical Report NDN-0001, Tech. Rep., 2010.