Empowering ICN as an Emerging Network Component for Frontier Applications

Hitoshi Asaeda
National Institute of Information and Communications Technology (NICT), Japan.
Email: asaeda@nict.go.jp

***Abstract*—Information-Centric Networking (ICN) has attracted significant attention as a network architecture optimized for efficient and scalable content dissemination. Moving beyond the conventional paradigm of name-based content retrieval, ICN can also support communication patterns associated with services and in-network computation. This paper aims to strengthen ICN by leveraging its core principles and introducing complementary mechanisms to improve efficiency, reliability, and performance within IP-based infrastructures. We present ICNx, an extension of the baseline ICN architecture that incorporates functional capabilities supporting adaptive and coordinated communication for both content delivery and function invocation. The study includes the architectural design and a prototype implementation on the open-source Cefore platform. In addition, a large-scale research initiative under Japan's JST Moonshot R&D program is discussed as an illustrative use case motivating the proposed design. The results demonstrate the practical feasibility of the approach and highlight the evolving role of ICN, not merely as a content-delivery mechanism, but as a foundational communication substrate for scalable and resilient applications.**

***Index Terms*—Information-Centric Networking, ICNx, CCNx, in-network computing, Cefore**

I. Introduction

Information-Centric Networking (ICN) represents a paradigm shift in network architecture by moving the focus of communication from host-based addressing to content-based retrieval [1], [2]. In contrast to conventional IP address-based communication, where data is accessed via a host's IP address, ICN retrieves content by name using "Interest" packets for requests and "Data" packets for replies. To support name-based forwarding, ICN maintains a Forwarding Information Base (FIB), which maps content names to outgoing interfaces for forwarding Interests, and a Pending Interest Table (PIT), which records incoming interfaces in order to forward Data packets along the reverse path. In ICN, an interface for forwarding packets is called a "Face."

By virtue of name-based communication, ICN enables content associated with a given name to be cached within the network, allowing subsequent requests to be satisfied by intermediate nodes rather than by the original content producer. This capability, commonly referred to as in-network caching, reduces content retrieval latency and alleviates server load. ICN also improves resource utilization by aggregating identical requests and by multicast forwarding of packets, which minimizes redundant traffic and improves overall network efficiency. Multipath forwarding is an intrinsic capability of ICN; it enables data delivery across multiple network paths, increasing throughput and enhancing resilience against link failures and congestion. ICN's security model is content-centric: data objects can be cryptographically signed using name-based schemes, ensuring authenticity and integrity regardless of the delivery path or content location. This model simplifies trust management in distributed and mobile environments where traditional host-based security models are less effective.

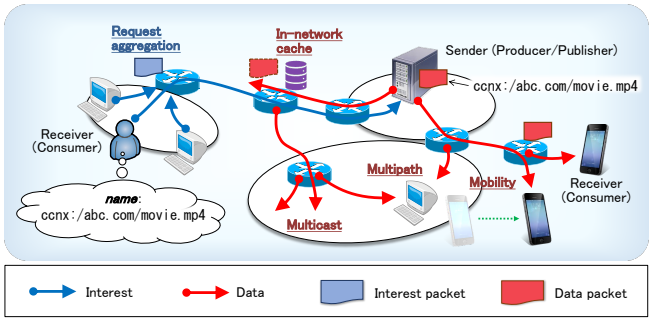

Fig. 1. Features natively supported in ICN (from [3]).

Early ICN research received notable attention as part of "clean slate" efforts to address limitations of the current Internet architecture. That line of inquiry encouraged novel concepts that anticipate future service requirements and environmental changes. However, the idea of entirely replacing the existing Internet infrastructure within a short timeframe is optimistic. Accordingly, momentum in ICN research is often perceived to have diminished.

Given the vast scale and entrenched deployment of the current IP-based Internet, a more pragmatic perspective treats ICN as a complementary architecture that can coexist and interoperate with existing systems. Recent work has increasingly focused on integrating ICN with current Internet services. As highlighted in a recent survey [3] and Fig. 1, ICN shows considerable potential to enhance scalability and efficiency, reinforcing its relevance as an evolutionary approach to future network design. These characteristics make ICN well-suited to real-time and distributed scenarios where responsiveness and decentralized control are essential. Immersive and large-scale distributed platforms, such as those envisioned for the metaverse, are promising application domains. Consequently, research efforts have focused on applying ICN to emerging services that require low-latency, high-throughput communication.

979-8-3315-7111-5/26/$31.00 ©2026 IEEE

In this paper, we consider ICN as a foundational network component capable of supporting a wide range of technologies by leveraging its intrinsic strengths, including in-network caching, multicast, and name-based service and function discovery. To meet the stringent requirements of adaptive, coordinated communication, we propose an extension of the baseline ICN architecture, referred to as ICNx, that introduces novel functional capabilities for collaborative and interactive communication. By utilizing a hop-by-hop communication paradigm, ICNx enables in-network execution of diverse functions and computational tasks, enhancing flexibility and efficiency in distributed environments. We introduce a prototype implementation using the open-source Cefore platform [4], [5] over IP networks and describe an illustrative use case drawn from Japan's JST Moonshot R&D program Goal 1 [6].

This paper is organized as follows. Section II provides background and motivation. Section III describes ICNx's core characteristics. Section IV presents preliminary experimental evaluation using Cefore. Section V describes example scenarios considered in our project. Section VI concludes and outlines future research directions.

II. ICN-Based In-Network Computing – Overview, Related Work, and Motivation

From a broader research perspective, ICN has gradually evolved into a complementary networking paradigm that enhances the capabilities of IP-based infrastructures. Its core mechanisms enable efficient data dissemination and facilitate application models that extend beyond conventional content retrieval. These architectural features are particularly well-suited to the demands of distributed and autonomous systems, enabling dynamic information exchange and coordination among heterogeneous entities.

Recent research has explored ICN's potential beyond content dissemination, particularly for in-network computing. For example, remote method invocation over ICN [10] provides a sophisticated framework in which consumers send a request packet with a function name and servers respond with a reply packet, enabling in-network invocation.

ICN-based service function chaining (SFC) mechanisms [11], [12], which execute multiple service functions for a content flow, have been explored to enable flexible in-network computing through the on-the-fly execution of an ordered set of functions required by a data flow, thereby supporting network-wide distributed computation. Hayamizu et al. [13] introduced an elastic ICN-based offloading framework that relocates service functions based on network load, demonstrating ICN's ability to integrate computation with forwarding. A remaining challenge is incorporating user-driven requirements into the offloading decision process.

Geng et al. [14] provided a systematization of knowledge (SoK) on distributed computing in ICN. They categorized approaches to orchestration, execution, and application of distributed tasks over ICN and highlighted open challenges in naming, caching, and resource coordination. While comprehensive, their work remains primarily descriptive and does not address higher-level semantic communication models or the architectural trade-offs encountered in concrete implementations.

Named Function Networking (NFN) represents another significant effort to integrate computing into ICN. Early work, such as [15], proposed expressing computations as named functions, enabling the resolution and caching of results within the network. Subsequent work introduced provenance mechanisms [16] to improve transparency. Although these contributions establish a strong foundation, challenges remain in optimizing function placement and in scaling complex function naming schemes.

In summary, prior work demonstrates that ICN can support computation through several techniques. This perspective highlights a new role for ICN, evolving from a content dissemination platform into a network component that enables in-network computing. Building on these studies, most existing approaches have focused on methods that closely align with core ICN principles. Meanwhile, emerging applications are beginning to expose requirements that extend beyond the traditional ICN design domain. Within this line of research, our recent work treated distributed artificial intelligence (DAI) tasks as named, invocable functions using ICN [17]. The proposed mechanism enables computational requests (e.g., object detection and anomaly analysis) to be routed and resolved based on intent-aware naming and in-network availability, allowing their execution to be supported by intermediate network nodes.

These observations suggest leveraging ICN beyond its traditional design domain, which in turn motivates the need for new architectural and functional extensions.

III. Extended Information-Centric Networking (ICNx)

In this section, we introduce ICNx, an architectural enhancement of ICN that supports adaptive, coordinated communication for emerging applications. ICNx preserves ICN's core principles while adding complementary mechanisms for collaborative and interactive communication, thereby improving efficiency, reliability, and performance. For a concrete ICNx protocol implementation, this study focuses on CCNx version 1.0 [18], [19], but the proposed functionality is equally applicable to Named Data Networking (NDN) [2].

A. Many-to-Many Communication

ICN inherently supports multicast communication, allowing multiple consumers to receive content simultaneously from a single producer. This model is *one-to-many multicast*, in which a single sender distributes data to multiple recipients. When multiple forwarders (i.e., producers or caching nodes) send data in response to an Interest from a single consumer, the consumer receives only the fastest-arriving Data packet. Subsequent responses from other forwarders are discarded. This behavior corresponds to an *anycast* model and is effective when in-network caches are used. Under current ICN design principles, Data returned for an Interest specifying a single

Algorithm 1 Interest Processing with Name and KeyIDs

Require: Interest packet $I = \{\text{Name } N, \text{KeyIDs } K = \{k_1, k_2, \dots\}\}$, Incoming Face f_{in}
Ensure: Forward Interest fragments to appropriate Faces

```
Retrieve FIB entry F for Name N
Initialize forwarding list L ← ∅
for each Face f in F do
    Determine KeyIDs K_f ← KeyIDs in K that are associated with Face f
    if |K_f| > 0 then
        Add (K_f, f) to L
if |L| = 0 then
    Drop Interest I
    return
for each (K_f, f) in L do
    Create PIT entry: {N, K_f, f_in}
    Forward Interest fragment {N, K_f} to Face f
```

Algorithm 2 Data Processing with Name and KeyIDs

Require: Data packet $D = \{\text{Name } N, \text{KeyID } k, \text{Payload}\}$
Ensure: Forward D to appropriate Faces and update PIT

```
Lookup PIT entries for Name N
if no PIT entry for N exists then
    Drop Data packet D
    return
Identify matching PIT entries where KeyID = k
if no matching entry exists then
    Drop Data packet D
    return
for each matching PIT entry (N, k, f_in) do
    Forward Data packet D to Face f_in
    Remove PIT entry (N, k, f_in)
Other PIT entries for N with different KeyIDs remain until their Data arrives
```

content name is unique; this assumption enables anycast forwarding because retrieving identical Data from multiple forwarders is unnecessary. However, if multiple distinct Data objects are associated with the same name, this assumption prevents *many-to-one communication*, in which a single consumer receives data from *all* available forwarders in response to a single Interest.

To support many-to-one communication, ICNx satisfies two key requirements: (1) an Interest that specifies a content name is forwarded to all potential producers via multipath routes, and (2) Data packets returned by those producers reach the consumer without aggregation by intermediate routers. The first requirement leverages ICN's inherent capability to propagate Interests to multiple producers. The second is addressed in ICNx through *explicit producer identification* using the "KeyID." KeyID, defined in the CCNx 1.0 specification, identifies the public key used to sign a CCNx message. It enables signature verification and key-based restrictions without transmitting the full key. In ICNx, KeyID additionally serves to indicate the target producer. Accordingly, ICNx routers maintain both FIB and PIT entries labeled by the pair of content name and KeyID, and remove PIT entries only after forwarding the matching Data packet. This design ensures that multiple Data packets for the same content name but different KeyIDs are correctly distinguished and delivered to the consumer. Algorithms 1 and 2 describe the Interest and Data forwarding with name and KeyID matching, respectively.

Consider the example shown in Fig. 2, which involves two consumers (C1 and C2) and four producers (P1–P4). Case (1) illustrates anycast: C1 sends an Interest containing the name ccnx:/robot/. All producers along the path that match this name return the corresponding Data; however, C1 receives only the first arriving Data packet, while subsequent Data packets are discarded. Case (2) illustrates one-to-many multicast: when C1 and C2 simultaneously issue Interests for the same name, ccnx:/robot/group-1, the Interests are aggregated en route to

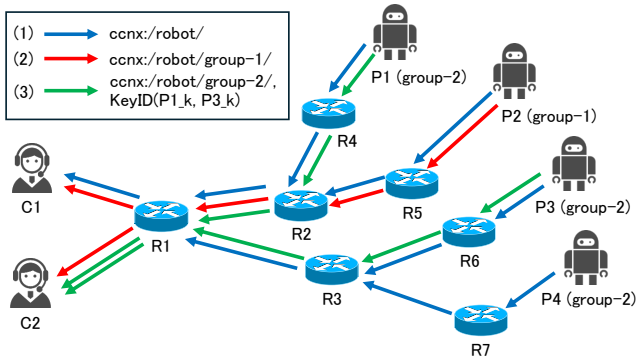

Fig. 2. Example scenarios illustrating anycast, one-to-many multicast, and many-to-one multicast. Arrow lines indicate Data forwarding paths, which are the reverse paths of Interest messages.

P2. The Data returned by P2 is duplicated at the branching router (R1) and delivered to both consumers. Case (3) illustrates many-to-one communication: C2 issues an Interest for the name /robot/group-2 that explicitly specifies the KeyIDs of P1 and P3, which are members of group-2. Upon receiving this Interest, P1 and P3 each return their respective Data. Unlike in Case (1), these Data packets are not discarded along the forwarding path and are successfully delivered to C2.

As described above, ICNx enables a consumer to collect data from multiple forwarders with a single Interest. Combined with ICN's standard one-to-many multicast, ICNx supports integrated bi-directional *many-to-many* communication.

B. Publish/Subscribe Communication

ICN enables two-way communication via Interest messages, which carry requests, and Data messages, which carry responses. The ICN protocol family is fundamentally a *pull-based* model. While pull-based communication is the primary mechanism, supporting *push-based* communication expands the range of feasible applications. These properties underpin responsive and scalable architectures and are essential for real-time and asynchronous applications. In this context,

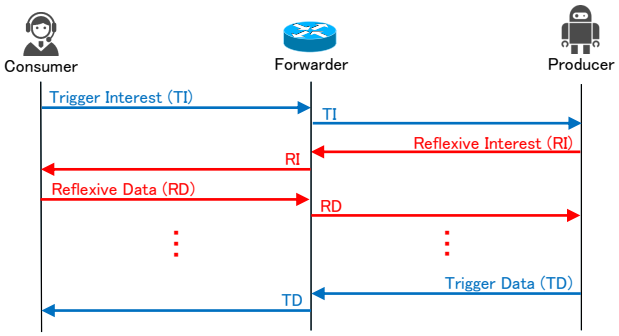

Fig. 3. Data flow of reflexive forwarding.

publish/subscribe (pub/sub) applications that combine pull and push paradigms offer an effective solution by providing the flexibility of on-demand access together with the immediacy of proactive updates.

Reflexive forwarding [20], recently proposed by the IRTF ICN Research Group, supports push-based communication and pub/sub applications while preserving ICN's core Interest–Data exchange model. Reflexive forwarding, therefore, fits naturally for seamless pub/sub integration within ICN.

Reflexive forwarding introduces two new Interest/Data types: "Trigger Interest/Data" and "Reflexive Interest/Data". As shown in Fig. 3, a consumer initiates an operation by sending a Trigger Interest (TI). The name specified in the TI is maintained in the FIB entry, and the TI eventually reaches the producer associated with that name. Upon receiving the TI, the producer sends a Reflexive Interest (RI). In response, the consumer sends the Reflexive Data (RD), which is the actual payload to be pushed. Once the producer has finished receiving data from the consumer, it sends a Trigger Data (TD) to notify the consumer that reception is complete. Large objects, such as video content, are segmented into chunks [21]; a producer may send multiple RIs for a single TI, and the consumer then replies with multiple RDs per RI. By leveraging these constructs, reflexive forwarding decouples pub/sub association from the routing protocol and eliminates the need for complex routing operations to maintain forwarding paths toward producers, significantly reducing the risk of FIB entry explosion.

Reflexive forwarding also handles scenarios in which a single consumer issues a TI for a name associated with multiple producers. Those producers return RIs to the consumer, and each producer subsequently receives RD from the consumer, enabling pub/sub communication from a single consumer to multiple producers. Combining the reflexive forwarding with the many-to-many multicast framework described earlier enables multiple consumers to receive data from multiple producers.

Reflexive forwarding further mitigates concerns about producer mobility raised in ICN research [22]. By treating the publisher node (i.e., consumer of reflexive forwarding) that pushes data and the subscriber node (i.e., producer of reflexive forwarding) that receives data, the subscriber can issue RI (triggered by TI) as a subscribe message and thus continue receiving data even if the publisher moves.

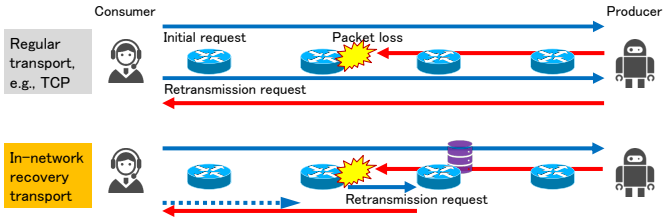

Fig. 4. ReBARC: Transport using in-network data recovery (presented in [23]).

C. In-network Data Recovery

In-network computing places processing inside the network, which can increase the risk of data loss because processing nodes are distributed, and network congestion may occur. To ensure reliable communication in such contexts, robust data recovery mechanisms are required to handle packet loss and to preserve data integrity within the network rather than relying solely on end-to-end recovery. To pursue this vision, we introduce a reliable data transfer mechanism called Recovery-Budget-Aware Retransmission Control (ReBARC) [23] that leverages in-network recovery techniques.

ReBARC adopts the concept of "Symbolic Interests (SMIs)" [24], which were proposed to enable the smooth fetching of real-time streaming data from a producer. SMIs allow wildcard specification of chunk numbers and support bulk data transfer. Although SMIs can reduce the risk of network congestion and packet loss in real-time streaming, consumers and intermediate nodes should quickly detect missing chunks and issue the regular Interests with chunk numbers to request retransmission of the missing chunks from upstream nodes.

ReBARC uses a *budget-aware* retransmission strategy that dynamically adjusts retransmission behavior based on the available recovery budget in the network. This strategy optimizes resource utilization while minimizing data loss. By enabling recovery from packet loss at intermediate nodes, ReBARC reduces the need for end-to-end retransmissions and improves overall communication reliability.

IV. Implementation and Preliminary Evaluations

We implemented the proposed functions on the open-source software platform called *Cefore* [4], [5], which is compliant with the CCNx-1.0 specification [18], [19]. Cefore comprises a forwarding daemon, *cefnetd*, and a content-store manager daemon, *csmgrd*, which provides in-network caching to *cefnetd* instances. *cefnetd* runs on Ubuntu, macOS, and Raspberry Pi OS; *csmgrd* runs on Ubuntu and macOS. Cefore also includes utilities such as *cefputfile* and *cefgetfile* for uploading and downloading Data, and tools such as *ccninfo* [7]. The platform exposes application programming interfaces (APIs) and a Python wrapper (*Cefpyco*) to simplify CCNx application development on Cefore.

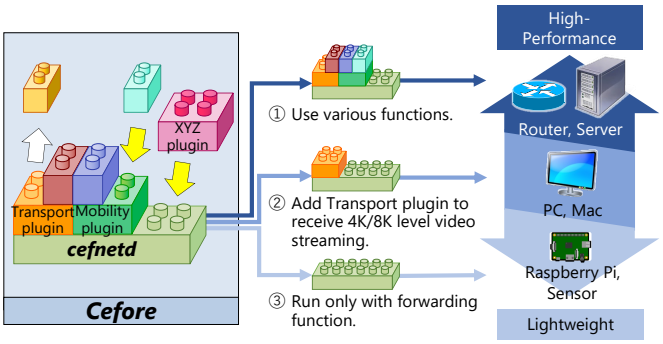

Fig. 5. Cefore software package (presented in [5]).

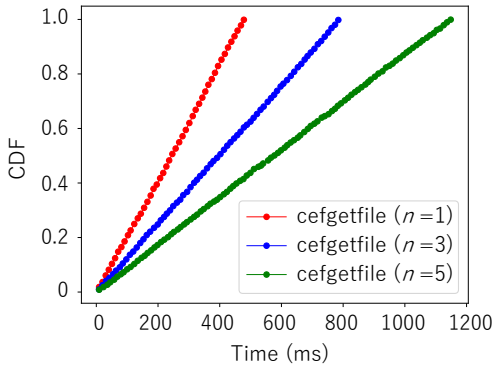

Fig. 6. Pull-based Data retrieval latency measured using the *cefgetfile* command.

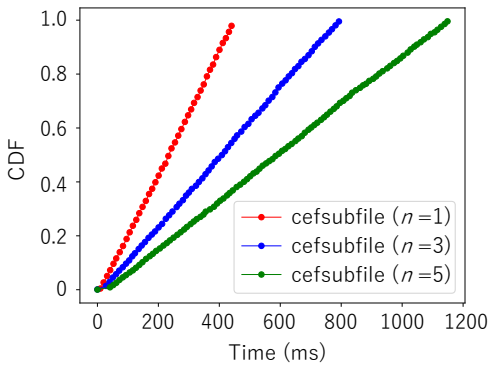

Fig. 7. Push-based Data delivery latency measured using the *cefsubfile* command.

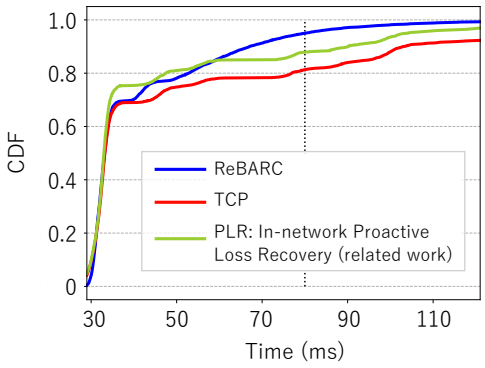

Fig. 8. End-to-end latency for streaming compared with Cefore/ReBARC and other approaches.

We implemented the functions described in Section III on top of Cefore and validated their behavior. We present measurements for two application models: a conventional pull-based Data retrieval using the *cefgetfile* command and a push-based Data delivery using the *cefsubfile* command based on reflexive forwarding. In both scenarios, a single consumer and a single producer are connected via intermediate routers that do not perform caching. We conducted both experiments by changing the number of intermediate routers (n) to 1, 3, and 5, as illustrated in Figs. 6 and 7, respectively. The content size was 30 MB, and its block size was 8 KB. The pipeline size was set to 32. All experiments ran in a Mininet emulator [25] running Cefore-0.11.0 on a PC with Ubuntu 22.04.

Under a minimal topology with a single intermediate router and sufficient underlying network bandwidth, our implementation achieves a throughput exceeding 500 Mbps with an average jitter of less than 150 μs. A key observation is that hop-by-hop forwarding introduces additional latency due to per-hop in-network processing at intermediate routers. To further improve performance and reliability, particularly in congested or complex network environments, transport mechanisms enabling rate-based congestion control (e.g., [26]) or techniques such as network coding (e.g., [27]) can be beneficial.

Additionally, we evaluated in-network recovery using ReBARC within the aforementioned Mininet emulator. Figure 8 reports results for a network topology with a single consumer and a single producer, each attached to an access point (AP); two intermediate routers interconnect two APs. Data streams triggered by SMIs were transmitted at 20 Mbps, and the recovery budget was set to 80 ms. In this experiment, ReBARC delivered over 90% of traffic within the 80-ms recovery budget and over 99% within 100 ms, while TCP and PLR (related work) did not meet these delay requirements.

V. JST Moonshot Research and Development Program

We have been participating in Japan's JST Moonshot R&D Program Goal 1 (hereafter, referred to as Moonshot project) [6], which envisions a society by 2050 in which humans are free from the limitations of body, brain, space, and time. The project aims to provide a *Cybernetic Avatar (CA)* infrastructure, a concept defined in [28], that extends human presence and agency beyond physical, temporal, and spatial limits. CA integrates physical and virtual avatars to enhance human capabilities across multiple dimensions, including cognition, perception, and mobility. CAs are intended to support diverse activities such as education, medical care, disaster response, factory operation, and social participation for elderly or physically challenged individuals. Within this project, we

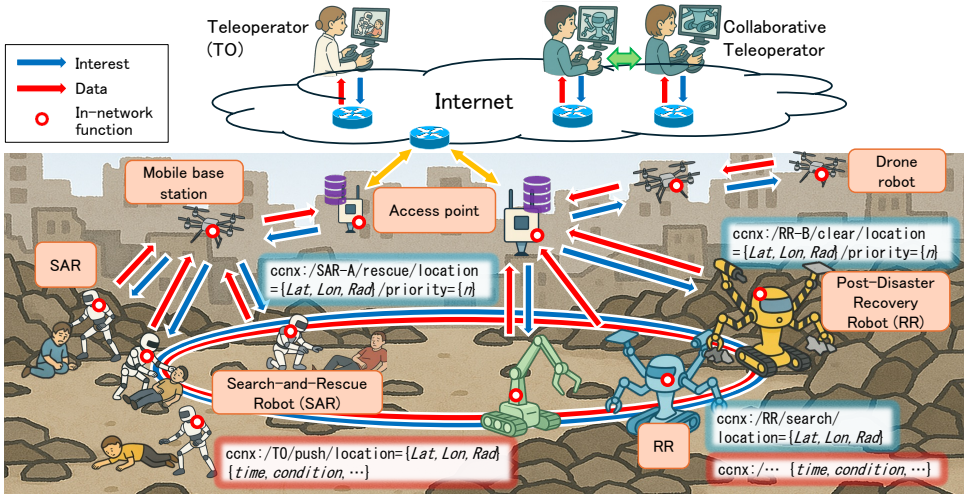

Fig. 9. Examples of Interest/Data exchange among TO, RR, and SAR entities (presented in [6]).

are developing a reliability-ensuring CA infrastructure that enables interactive teleoperation via ICNx.

In heterogeneous environments where conditions change dynamically and frequently, providing flexible and efficient data sharing is crucial for effective coordination among users or nodes. Obsolete information, often referred to as Age-of-Information (AoI), can lead to incorrect operational decisions. ICNx addresses these challenges by supporting synchronous, low-latency communication and significantly reducing bandwidth consumption via multicast among users or nodes that share common data streams, which is particularly beneficial under unstable network conditions. Furthermore, the ICNx platform supports seamless interaction and coordination among users or nodes through a pub/sub scheme, anchor-less mobility, and effective operation in serverless environments.

One potential use case considered in the Moonshot project is disaster response. CA technologies can support various *disaster response and recovery robots* in disaster-stricken areas. These CA robots collaborate with teleoperators (TOs) via ICNx-enabled, low-latency, and reliable communication. Figure 9 illustrates that robots assist with urgent tasks, such as search-and-rescue operations (Search-and-Rescue Robots, SAR), and later contribute to recovery tasks, such as debris removal and infrastructure restoration (Post-Disaster Recovery Robots, RR).

Figure 9 illustrates several representative examples of Interest/Data interactions. When a TO issues an Interest with the name format, ccnx:/RR/search/location={*Lat,Lon,Rad*}, all RRs associated with the prefix ccnx:/RR receive the Interest. RRs located within the specified geographic area respond by sending corresponding Data packets that include information such as {*time,condition,...*} via reverse paths.

Subsequently, the TO can actuate a specific Search and Rescue Agent (SAR)-A to perform rescue operations by sending an Interest of the form, ccnx:/SAR-A/rescue/location={*Lat,Lon,Rad*}/priority={*n*}. Upon receiving this Interest, SAR-A returns a Data packet as an acknowledgment and initiates the requested rescue behavior.

SAR units can also proactively report (push) status by issuing a Trigger Interest toward TOs, using the format, ccnx:/TO/push/location={*Lat,Lon,Rad*}{*time,condition,...*}, as described in Section III-B. In response, the TO sends Reflexive Interests to obtain the latest condition information (Reflexive Data) from the SAR units.

VI. Conclusion

ICN's reorientation toward content and name-based communication yields several architectural advantages, including in-network caching, multicasting, multipath forwarding, and embedded security mechanisms. These features collectively improve network performance, scalability, and flexibility.

In this paper, we advocate empowering ICN by treating it as a foundational component of current and future network services. We proposed ICNx, an extension that builds on ICN principles and adds mechanisms for many-to-many communication, pub/sub interaction, and in-network data recovery. These extensions increase service flexibility and efficiency in distributed environments. We also presented a prototype implementation using Cefore and introduced a reliability-ensuring CA infrastructure enabling interactive teleoperation via ICNx as an illustrative use case.

Building on ongoing theoretical research and system implementation efforts in ICN, including reliable communication protocols and semantic communication, we will advance ICNx through both fundamental studies and practical experimentation. In parallel with these research activities, the Moonshot project provides a platform for conducting large-scale validation and demonstration experiments, applying ICNx to real-world social systems in collaboration with multiple partners.

Acknowledgment

This work was partly supported by JST Moonshot R&D, Goal 1, Grant Number JPMJMS2216. The author would like to thank Yusaku Hayamizu, Htet Htet Hlaing, and Kazuhisa

Matsuzono for their valuable comments and assistance with implementation and experiments.

REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking Named Content," *Proc. ACM CoNEXT*, Dec. 2009.

[2] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, K. Claffy, P. Crowley, C. Papadopoulos, L. Wang, and B. Zhang, "Named data networking," *ACM Comput. Commun. Rev*., vol. 44, no. 3, 2014.

[3] H. Asaeda, K. Matsuzono, Y. Hayamizu, H. H. Hlaing, and A. Ooka, "A Survey of Information-Centric Networking: The Quest for Innovation," *IEICE Trans. Commun.*, Vol.E107-B, No.1, Jan. 2024.

[4] "Cefore," Available at: https://github.com/cefore/, accessed Nov. 25, 2025.

[5] H. Asaeda, A. Ooka, K. Matsuzono, and R. Li, "Cefore: Software Platform Enabling Content-Centric Networking and Beyond," *IEICE Trans. Commun.*, Vol.E102-B, No.9, pp.1792-1803, Sep. 2019.

[6] "Moonshot Goal 1: Realization of a society in which human beings can be free from limitations of body, brain, space, and time by 2050," Available at: https://www.jst.go.jp/moonshot/en/program/goal1/, accessed Dec. 1, 2025.

[7] H. Asaeda, A. Ooka, and X. Shao, "CCNinfo: Discovering Content and Network Information in Content-Centric Networks," IRTF RFC 9344, Feb. 2023.

[8] J. Ren, H. Guo, C. Xu, Y. Zhang, "Serving at the Edge: A Scalable IoT Architecture Based on Transparent Computing," *IEEE Network*, Vol. 31, No. 5, 2017, pp. 96–105.

[9] M. Sifalakis, B. Kohler, C. Scherb, and C. Tschudin, "An Information Centric Network for Computing the Distribution of Computations," *Proc. ACM ICN*, Sep. 2014, Paris, France.

[10] M. Król, et al., "RICE: Remote Method Invocation in ICN," *Proc. ACM ICN*, Sep. 2018, Boston, USA.

[11] M. Arumaithurai, J. Chen, E. Monticelli, X. Fu, and K. K. Ramakrishnan, "Exploiting ICN for Flexible Management of Software-Defined Networks," *Proc. ACM ICN*, Sep. 2014, Paris, France.

[12] L. Liu, et al., "ICN-FC: An Information-Centric Networking Based Framework for Efficient Functional Chaining," *Proc. IEEE ICC*, 2017, Paris, France.

[13] Y. Hayamizu, K. Matsuzono, T. Hirayama, and H. Asaeda, "Design and Implementation of ICN-Based Elastic Function Offloading Network for SFC," *Proc. IEEE NetSoft*, June 2021, Online.

[14] W. Geng, Y. Zhang, D. Kutscher, A. Kumar, S. Tarkoma, and P. Hui, "SoK: Distributed Computing in ICN," arXiv, 2023.

[15] M. Król and I. Psaras, "NFaaS: Named Function as a Service," *Proc. ACM ICN*, Sep. 2017, Berlin, Germany.

[16] C. Marxer and C. Tschudin, "Result Provenance in Named Function Networking," *Proc. ACM ICN*, Sep. 2020, Montreal, Canada.

[17] H. H. Hlaing and H. Asaeda, "Inference by Name: Adaptive In-Network Task Execution for Low-Latency Distributed Intelligence," *Proc. IEEE Globecom*, Dec. 2025, Taiwan.

[18] M. Mosko, I. Solis, and C. Wood, "Content-Centric Networking (CCNx) Semantics," IRTF, RFC 8569, July 2019.

[19] M. Mosko, I. Solis, and C. Wood, "Content-CentricNetworking(CCNx) Messages in TLV Format," IRTF, RFC 8690, July 2019.

[20] D. Oran, D. Kutscher, H. Asaeda, and K. Calvert, "Reflexive Forwarding for CCNx and NDN Protocols," IRTF Internet draft (work in progress), Sep. 2025.

[21] M. Mosko and H. Asaeda, "CCNx Content Object Chunking," IRTF Internet draft (work in progress), Oct. 2025.

[22] J. Augé, G. Carofiglio, G. Grassi, L. Muscariello, G. Pau, and X. Zeng, "MAP-Me: Managing Anchor-Less Producer Mobility in Content-Centric Networks," *IEEE TNSM*, Volume 15, Issue 2, June 2018.

[23] K. Matsuzono and H. Asaeda, "ReBARC: Recovery-Budget-Aware In-Network Retransmission Control in ICN," *Proc. IFIP Networking*, May 2025, Limassol, Cyprus.

[24] K. Matsuzono and H. Asaeda, "NMRTS: Content Name-Based Mobile Real-Time Streaming," *IEEE Communications Magazine*, Vol.54, No.8, Aug. 2016.

[25] "Mininet", Available at: http://mininet.org/, accessed Nov. 25, 2025.

[26] K. Schneider, C. Yi, B. Zhang, and L. Zhang, "A Practical Congestion Control Scheme for Named Data Networking," *Proc. ACM ICN*, Sep. 2016.

[27] K. Matsuzono, H. Asaeda, and T. Turletti, "Low Latency Low Loss Streaming Using In-Network Coding and Caching," *Proc. IEEE INFOCOM*, May 2017, Atlanta, USA.

[28] N. Hagita, et al., "Cybernetic avatars: Teleoperation technologies from in-body monitoring to social interaction," *Science Robotics*, Vol.8, No.74, 2023.