Content-based Filtering Discovery Protocol (CFDP): Scalable and Efficient OMG DDS Discovery Protocol

8th ACM DEBS 2014
May 28, 2014, Mumbai, India

Kyoungho An, Aniruddha Gokhale, and Douglas Schmidt
Dept. of EECS, Vanderbilt University

Sumant Tambe, Paul Pazandak, and Gerardo Pardo-Castellote
Real-Time Innovations (RTI)

Data Distribution Service (DDS)

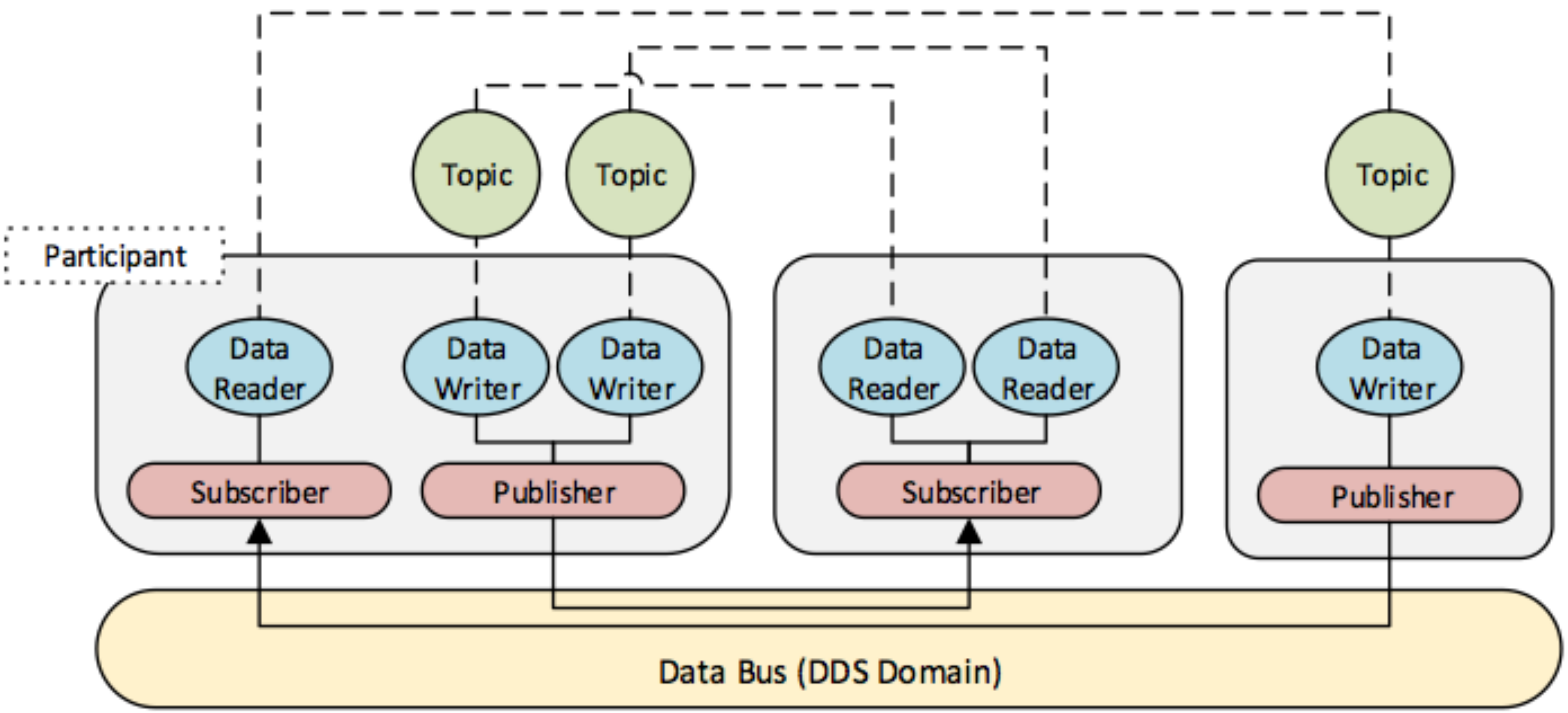
- DDS is an OMG standard specification for data-centric publish/subscribe middleware

- DDS is deployed in many IoT application domains, including Aerospace & Defense, Healthcare, Energy, Transportation, Control Systems
  - Interoperability
  - Scalability
  - QoS support

DDS Elements & Features

- Domain
- Participant
- DataWriter (DW) and DataReader (DR)
- Topic
- Quality-of-service (QoS)
- Content Filtered Topic (CFT)

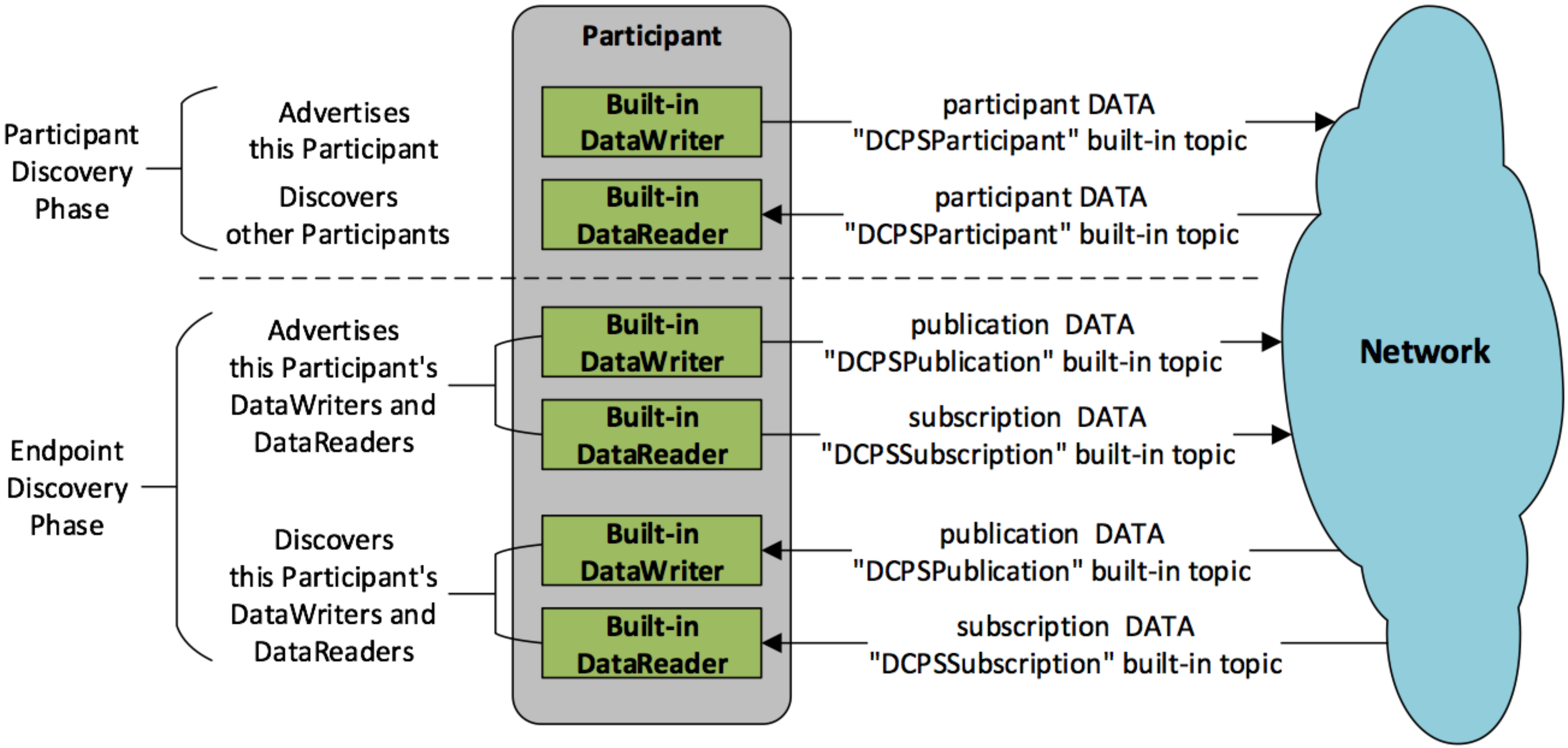
DDS Architecture



DDS Discovery Protocol

- OMG DDS Real-Time Publish-Subscribe (RTPS) defines a discovery protocol
  - Participant Discovery Protocol (PDP)
  - Endpoint Discovery Protocol (EDP)
- The DDS RTPS specification describes Simple Discovery Protocol (SDP) as a default discovery protocol
  - Simple Participant Discovery Protocol (SPDP)
  - Simple Endpoint Discovery Protocol (SEDP)

DDS Discovery Protocol Entities



Problem?

- SDP scales poorly as the number of peers and their endpoints increases in a domain
- Why?
  - Each peer sends/receives discovery messages to/from other peers in the same domain
- For a large scale system, substantial network, memory, and computing resources are consumed just for the discovery process
- This overhead degrades discovery completion time and hence overall scalability

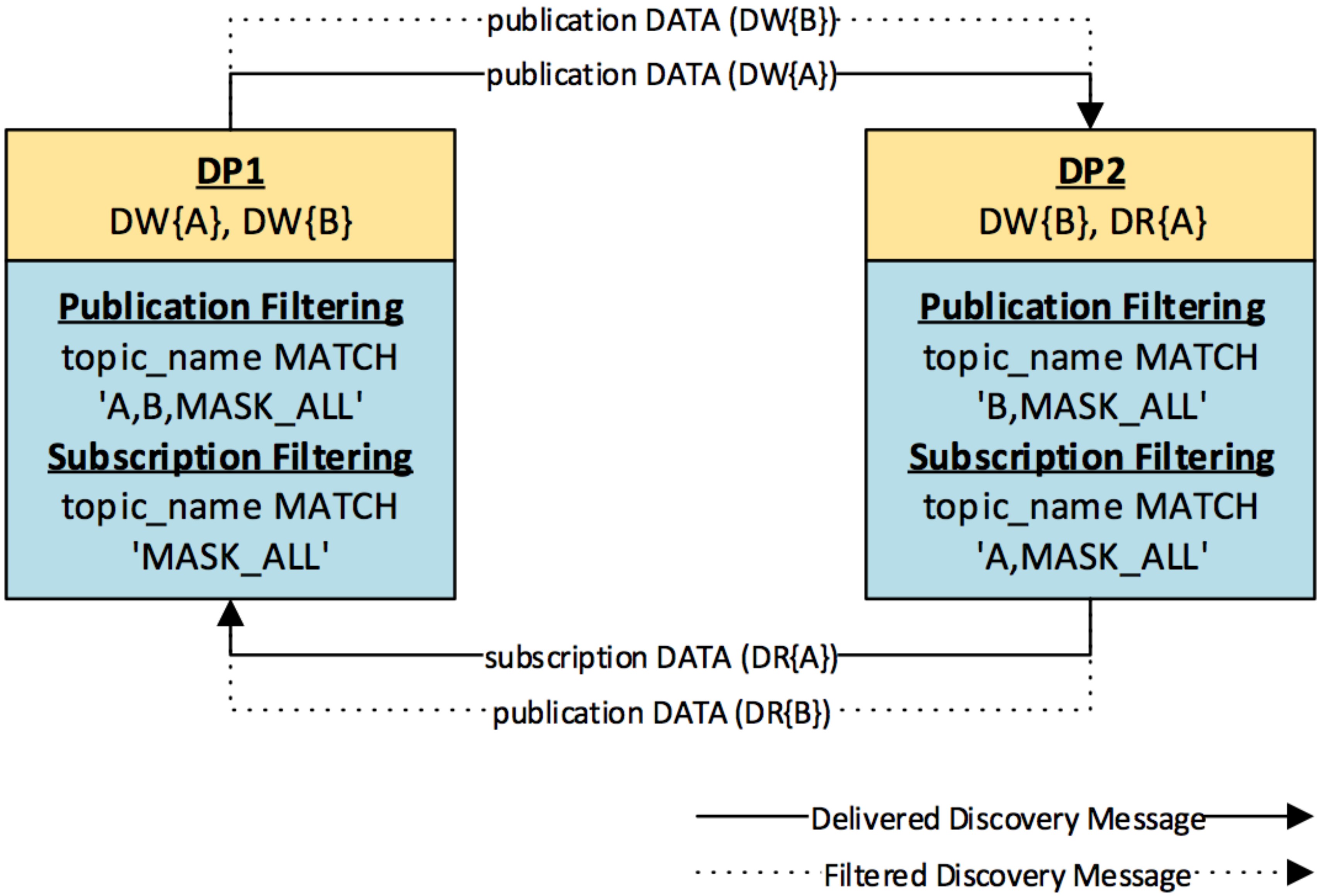
Solution

- A new mechanism for scalable DDS discovery called the Content-based Filtering Discovery Protocol (CFDP)
- CFDP employs content-based filtering on the sending peers to filter out unnecessary discovery messages by exchanging filtering expressions that limit the range of interests
- For the implementation, CFDP uses Content Filtered Topic (CFT) for built-in discovery entities

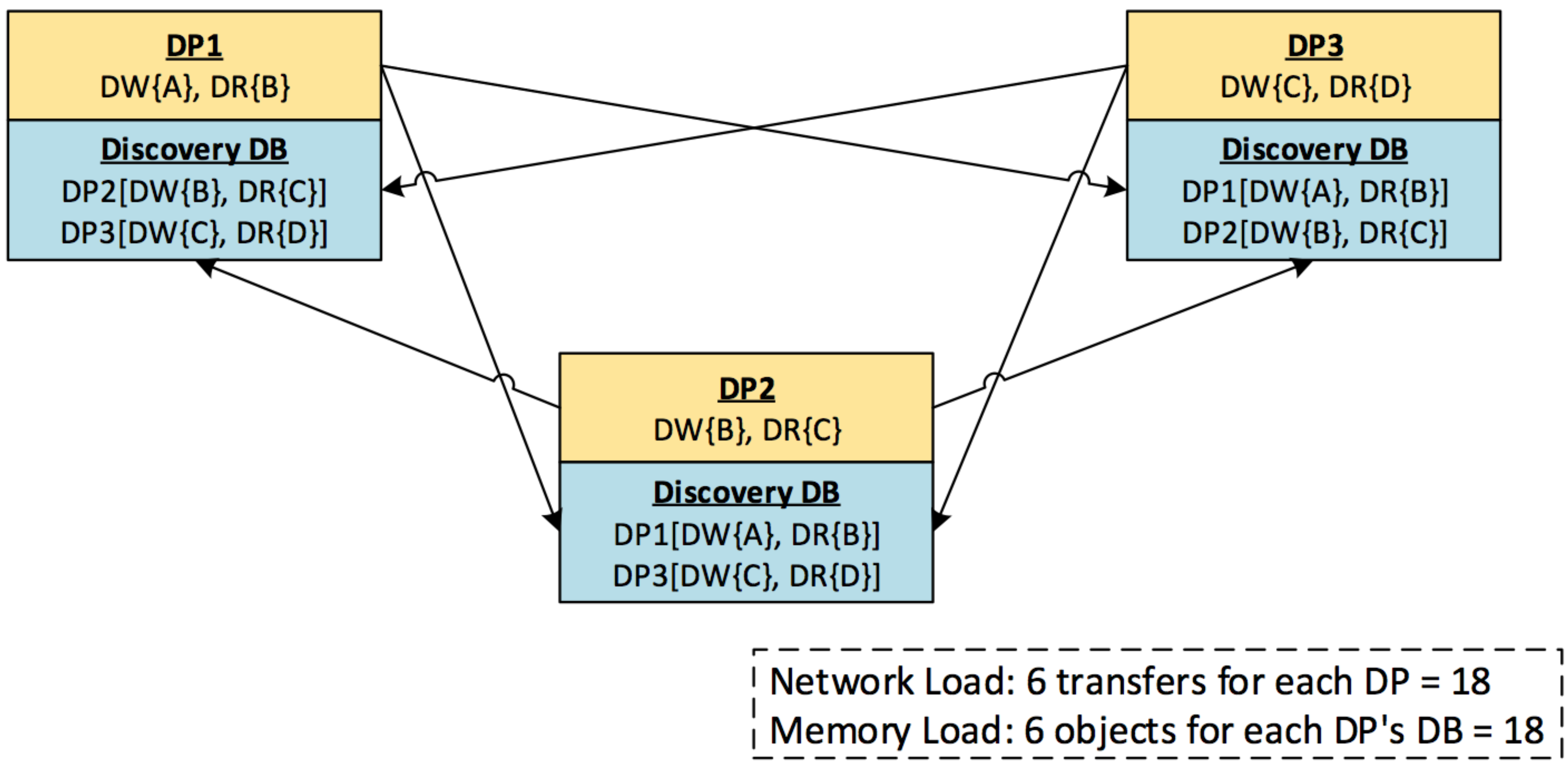
Design of CFDP

- CFDP filters out discovery messages based on topics names and endpoint types
- Utilizes the first phase of SDP called SPDP
- Differs from SDP in the endpoint discovery phase
- Uses built-in entities with CFTs that filter discovery messages on topic names stored in subscription DATA and publication DATA

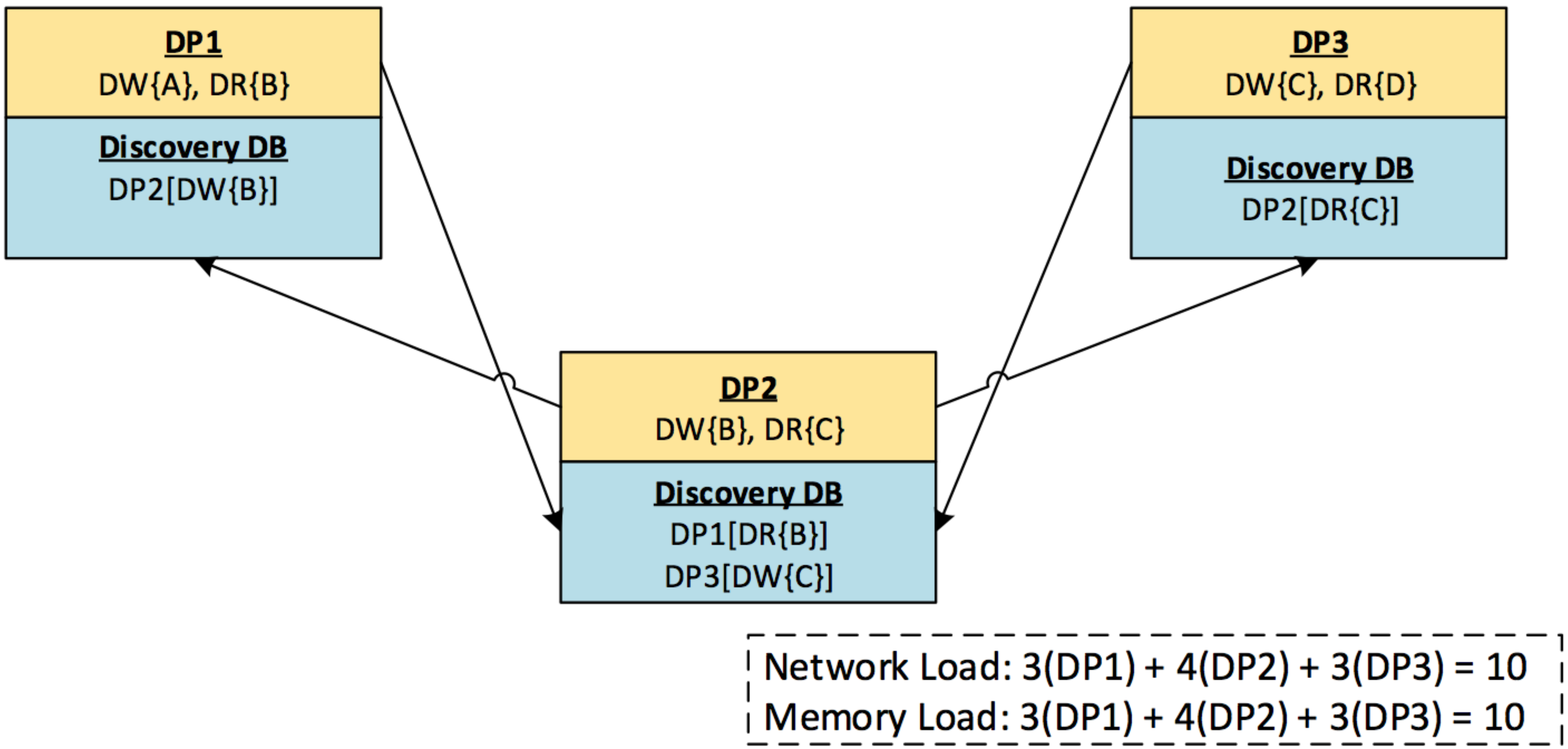
Design of CFDP



SDP Example


DP1
DW{A}, DR{B}
Discovery DB
DP2[DW{B}, DR{C}]
DP3[DW{C}, DR{D}]
DP3
DW{C}, DR{D}
Discovery DB
DP1[DW{A}, DR{B}]
DP2[DW{B}, DR{C}]
DP2
DW{B}, DR{C}
Discovery DB
DP1[DW{A}, DR{B}]
DP3[DW{C}, DR{D}]
Network Load: 6 transfers for each DP = 18
Memory Load: 6 objects for each DP's DB = 18

CFDP Example



CFDP Implementation

- CFDP prototype is implemented at the application-level to avoid non-standard changes to the underlying DDS middleware
- Use provided callback functions (Pluggable Discovery Framework) to exploit required events from the middleware
  - Creation/Deletion events of local DDS entities
- Create built-in DDS entities to exchange discovery events with remote peers
  - Creation/Deletion events of remote DDS entities

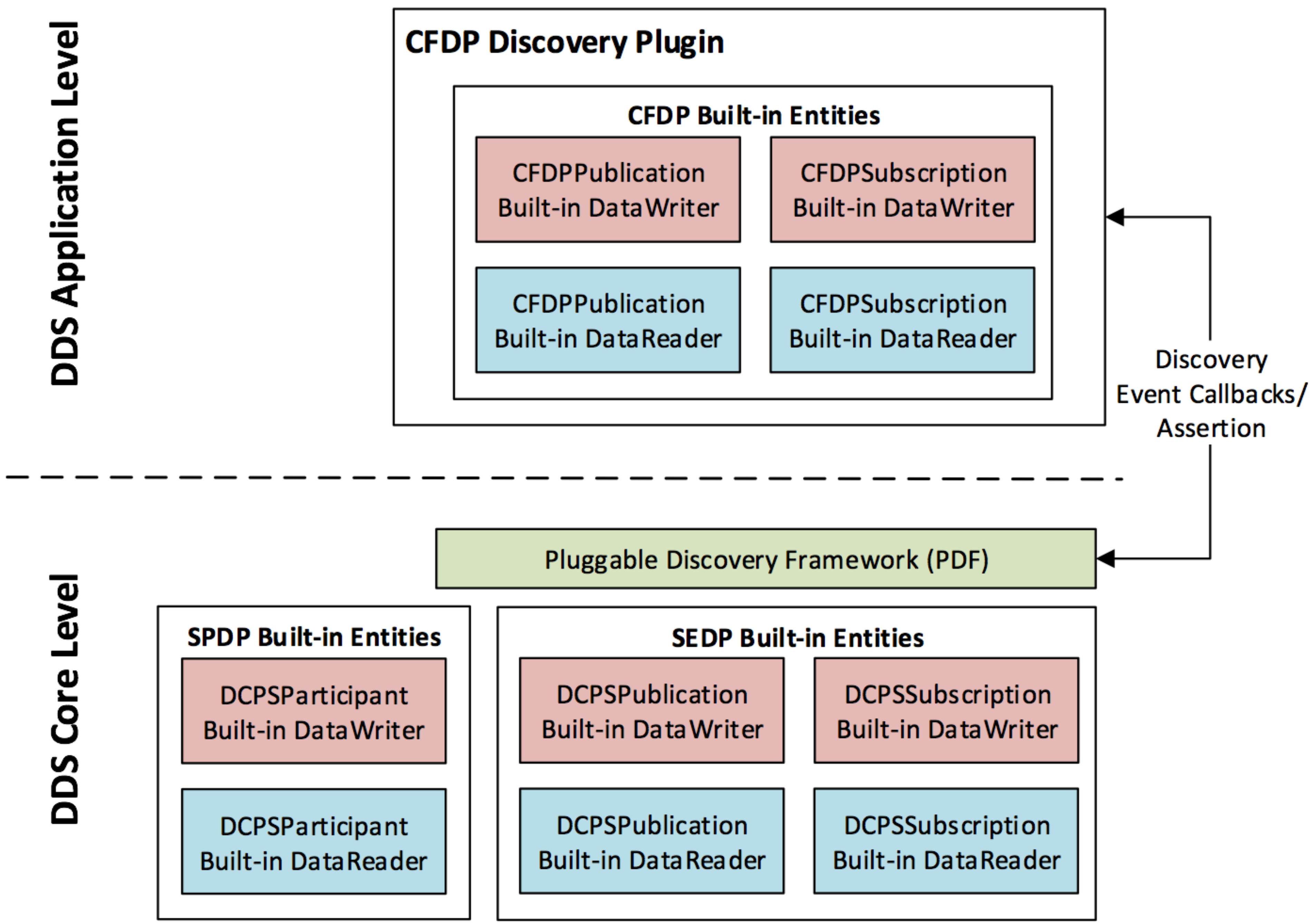
Assumption by Practical Limitation

- Our prototype implementation assumes users determine which topics are published or subscribed by participants when participants are created

- CFDP uses TRANSIENT_LOCAL durability QoS for late joiners, but it does not work properly with using CFT

- When a value in a filtering expression of a CFT is changed, the changed value is not reflected in the list of late joiners

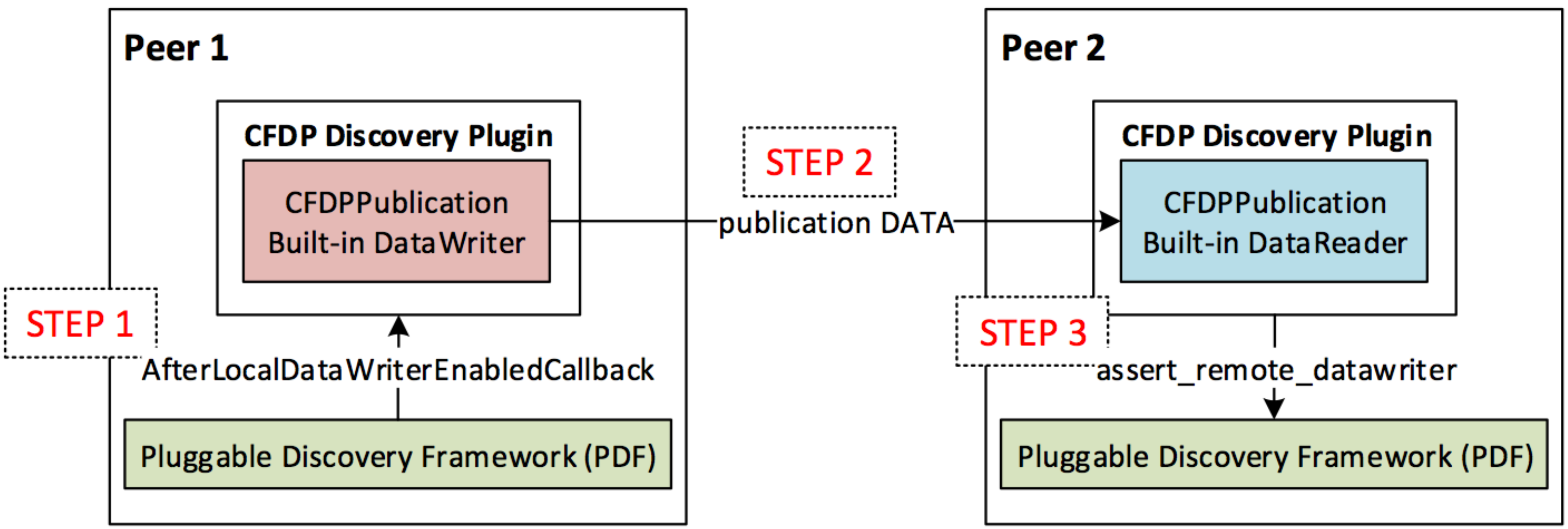
CFDP Software Architecture



CFDP Callback Functions

- Pluggable Discovery Framework (PDF)
  - Local Endpoint Enabled
  - Local Endpoint Deleted
- Built-in Entities
  - Remote DataWriter Received
  - Remote DataReader Received

CFDP Sequence of DW Creation Event



SDP Network Usage with Multicast

$$N_{multi\_participant} = \frac{E}{P} + \frac{E}{P} \cdot (P - 1) \quad (1)$$

$$= \frac{E}{P} + E - \frac{E}{P} \quad (2)$$

$$= E \quad (3)$$

$$N_{multi\_total} = E \cdot P \quad (4)$$

CFDP Network Usage with Multicast

$$N_{multi\_participant} = \frac{E}{P} + \frac{E}{P} \cdot (P - 1) \cdot R \tag{12}$$

$$= \frac{E}{P} + E \cdot R - \frac{E}{P} \cdot R \tag{13}$$

$$= F \cdot (1 - R) + E \cdot R \tag{14}$$

$$\sim E \cdot R \tag{15}$$

$$N_{multi\_total} \sim E \cdot P \cdot R \tag{16}$$

SDP Network Usage with Unicast

$$N_{uni\_participant} = \frac{E}{P} \cdot (P-1) + \frac{E}{P} \cdot (P-1) \quad (5)$$

$$= 2 \cdot \frac{E}{P} \cdot (P-1) \quad (6)$$

$$\because (P-1) \sim P \quad (7)$$

$$\sim 2 \cdot E \quad (8)$$

$$N_{uni\_total} \sim 2 \cdot E \cdot P \quad (9)$$

CFDP Network Usage with Unicast

$$N_{uni\_participant} = \frac{E}{P} \cdot (P-1) \cdot R + \frac{E}{P} \cdot (P-1) \cdot R \quad (17)$$

$$= 2 \cdot \frac{E}{P} \cdot (P-1) \cdot R \quad (18)$$

$$\sim 2 \cdot E \cdot R \quad (19)$$

$$N_{uni\_total} \sim 2 \cdot E \cdot P \cdot R \quad (20)$$

SDP Memory Usage

$$M_{participant} = E \tag{10}$$

$$M_{total} = E \cdot P \tag{11}$$

CFDP Memory Usage

$$M_{participant} = \frac{E}{P} + \frac{E}{P} \cdot (P - 1) \cdot R \tag{21}$$

$$\sim E \cdot R \tag{22}$$

$$M_{total} \sim E \cdot P \cdot R \tag{23}$$

Empirical Evaluation

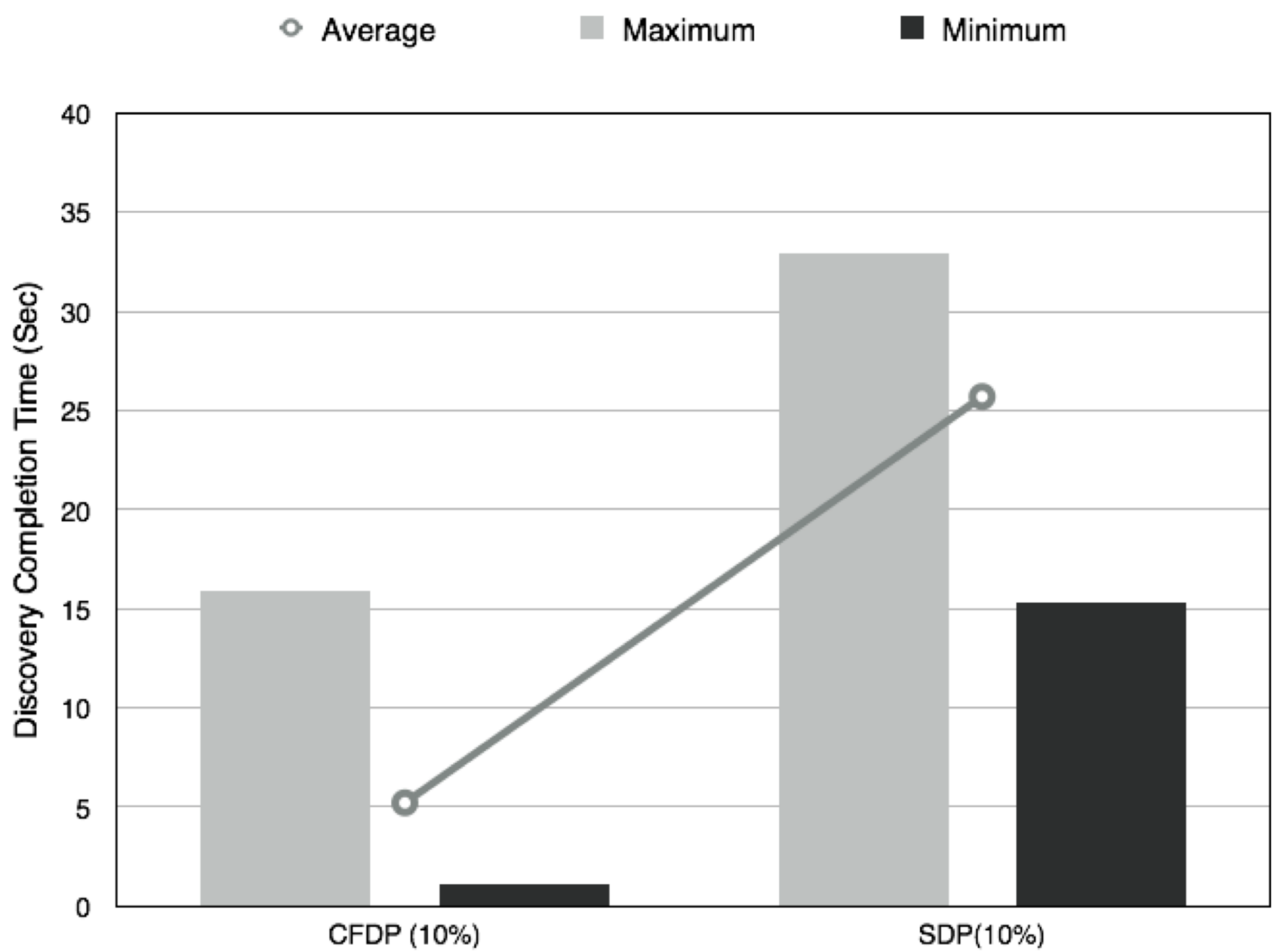
- Discovery Completion Time
  - CFDP vs. SDP (10% matching)
- CPU Usage
  - CFDP vs. SDP (10% matching)
  - CFDP (10%, 30%, 50%)
- Memory & Network Usage
  - CFDP vs. SDP (10%, 50%, 100%)

Empirical Evaluation

- Testbed
  - Six 12-core machines
  - 1Gb Ethernet connected to a single network switch
  - RTI Connext DDS 5.0
- Experiment Setup
  - 480 applications (participants)
  - Each participant has 20 endpoints
  - Default matching ratio is 0.1 (10%)
  - SDP uses multicast and CFDP uses unicast

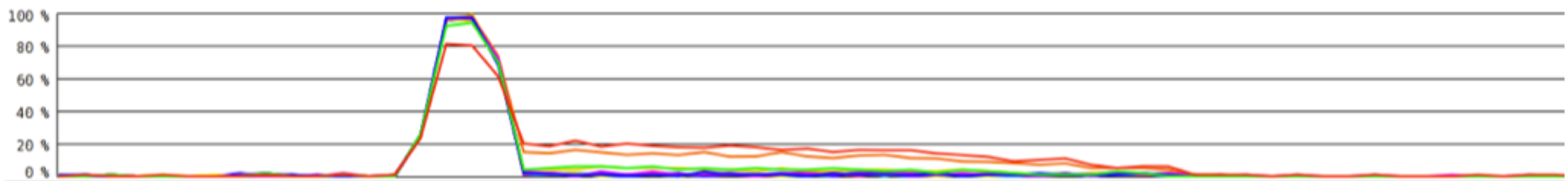
Discovery Completion Time

- Discovery completion time is defined as the time needed to completely discover all matching endpoints in a domain

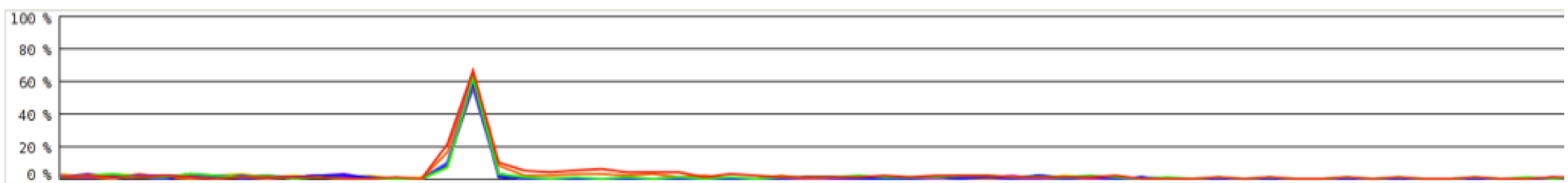


CFDP and SDP CPU Usage

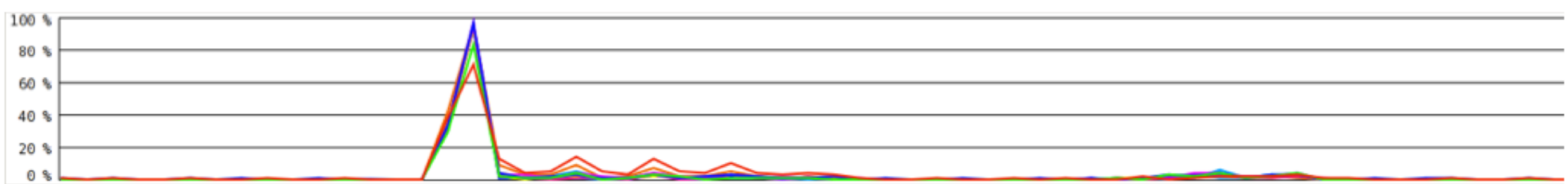
SDP CPU Usage (10% Matching)



CFDP CPU Usage (10% Matching)

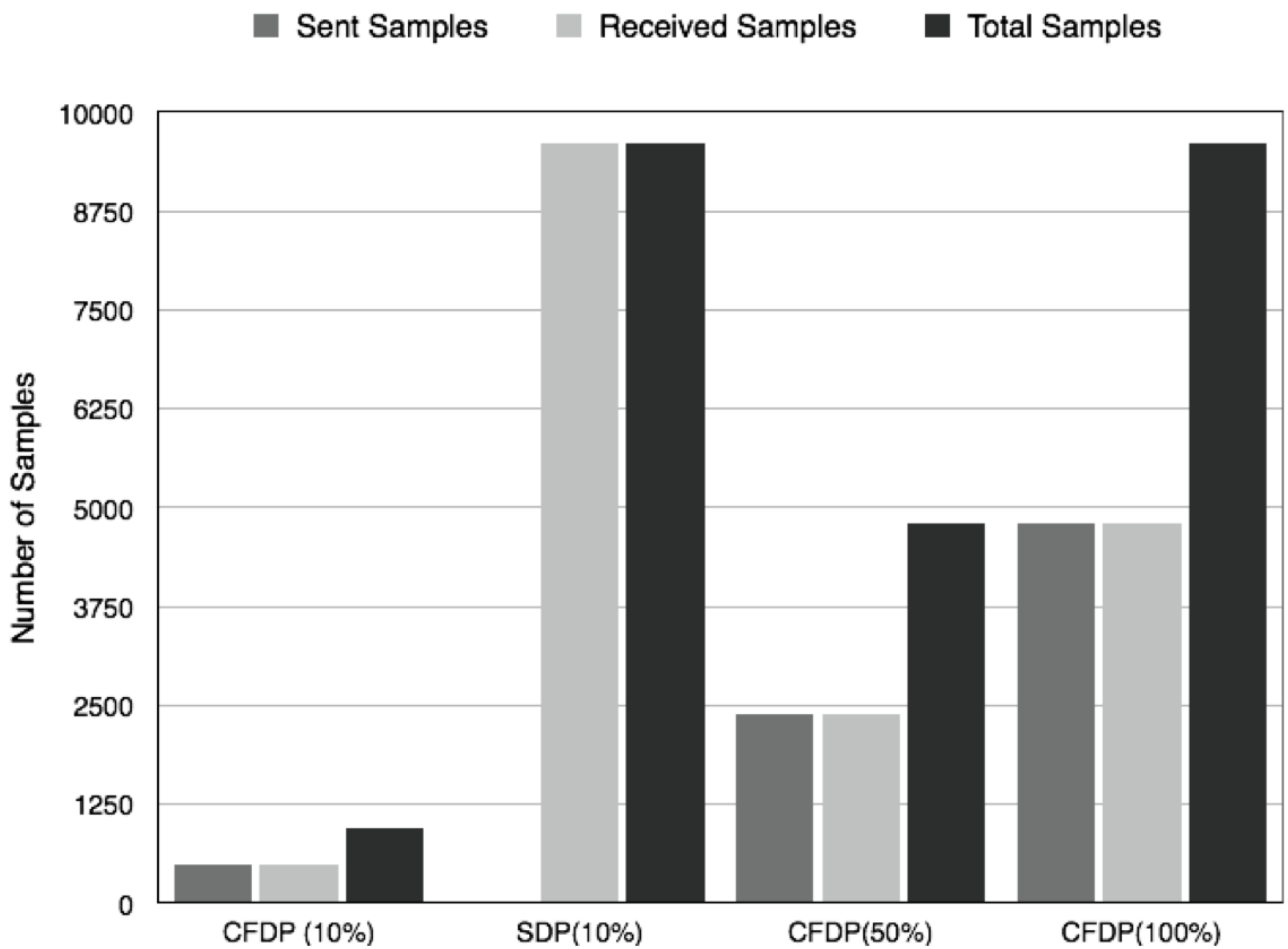


CFDP CPU Usage (30% Matching)



CFDP CPU Usage (50% Matching)

Sent/Received Discovery Messages



Discussion

- CFDP is more efficient and scalable than SDP
- CFDP's current lack of support for multicast can impede scalability
- Instance-based filtering can help to make CFDP scalable in a large-scale system with a small set of topics

Questions?