A Knowledge-based Approach for Resource Discovery and Allotment in Swarm Middleware

M. **Ruta**, F. **Scioscia**, E. **Bove**,
A. **Cinquepalmi**, E. **Di Sciascio**

Politecnico di Bari, Bari, Italy

SET-222 - Specialists Meeting on Swarm centric Solution for Intelligent Sensor Networks
June 7th-8th, 2016 – Rome, Italy

Outline

- Service-Oriented Architecture for the Internet of Things
  - State of the art and issues
- Semantic-based Swarm Middleware Architecture
  - Bee Data Distribution Service
  - Ubiquitous Knowledge Base model
  - Semantic-based service/resource discovery
- Experiments
- Conclusion and Future Work

Service-Oriented Architecture

- SOA is a scalable paradigm in military and civil domains for service
  - interoperability
  - flexibility
  - reusability
- NATO research task group IST-118 provides best practices to make SOA applicable to the area of NATO Command, Control, Communications, Computers, Intelligence, Surveillance and Reconnaissance (C4ISR)

SOA for Internet of Things

ISSUES

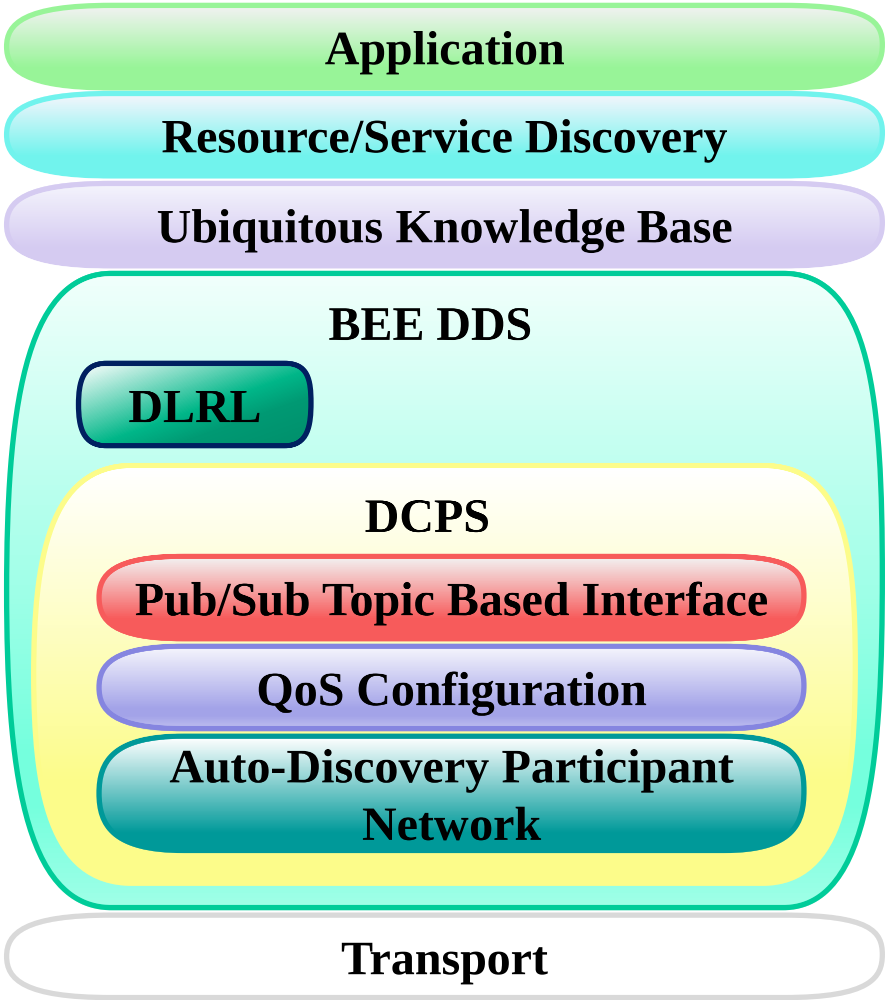
- Computational resource scarcity
- Device mobility
- Service volatility
- Network unreliability

IDEAS

- Dissemination protocols in decentralized systems
- Lightweight semantic swarm middleware
  - based on reliable data distribution systems
- Dynamic, autonomous, distributed service/resource discovery

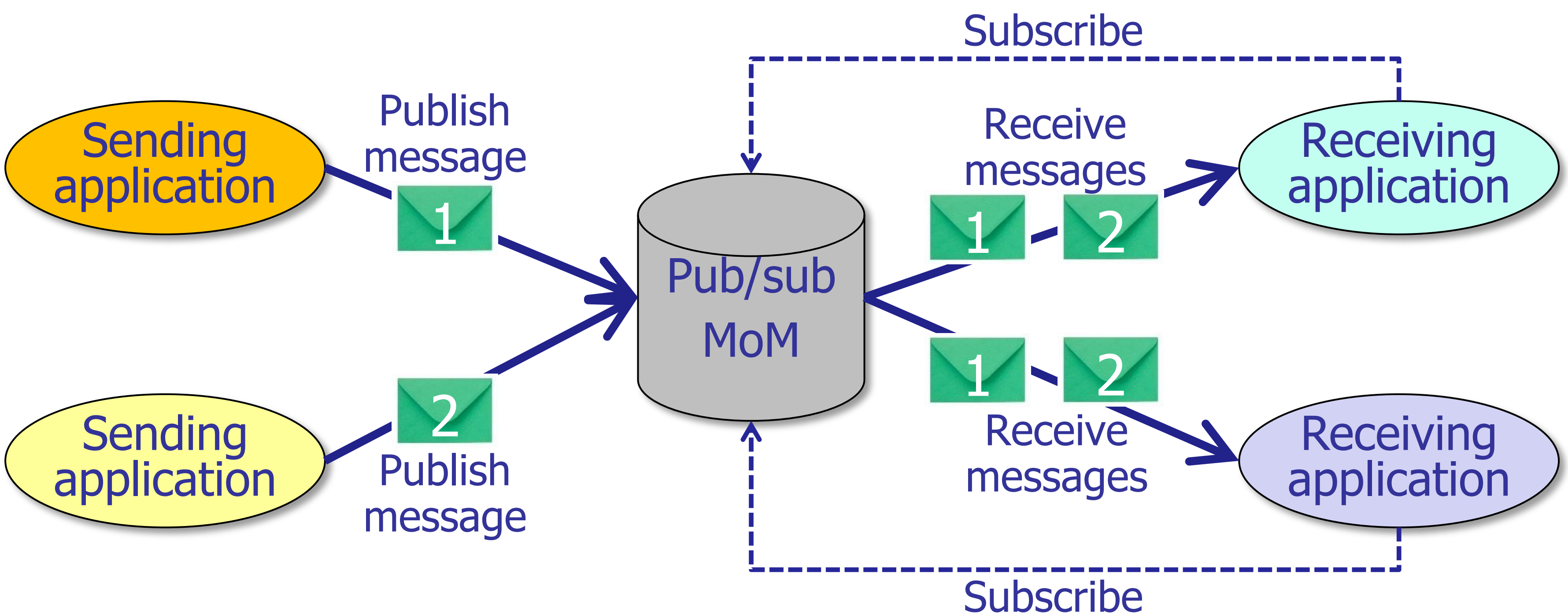
Semantic-based Swarm Middleware Architecture

- Knowledge-based approach for resource discovery, allocation and sharing in distributed swarm scenarios
- Functional layers on top of a standard publish/subscribe Message-Oriented Middleware



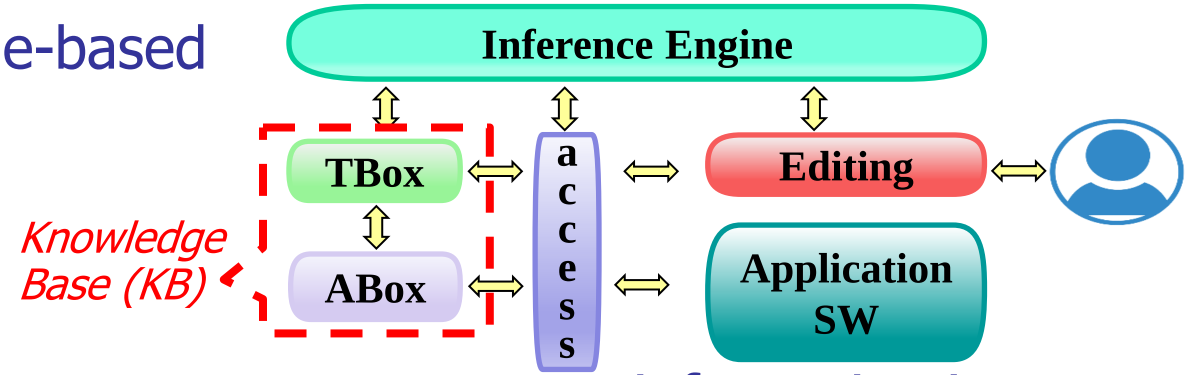
Publish/subscribe Message-Oriented Middleware

- One-to-many distribution of messages
- Messages marked by topics
- Messages received by all subscribers of the related topic



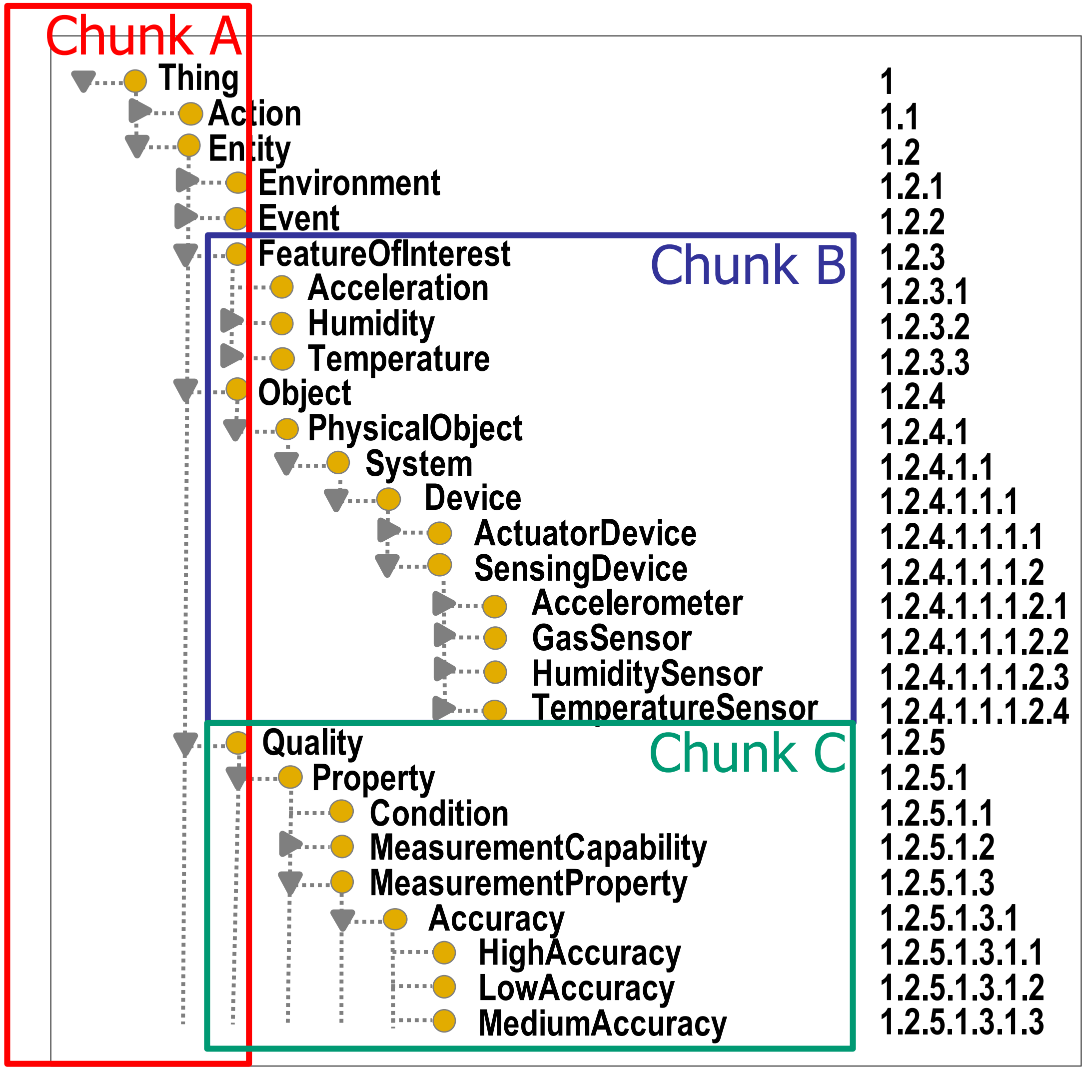
Ubiquitous Knowledge Base (1/2)

- Classical knowledge-based systems:



- u-KB: layer granting transparent access to information in a swarm of semantic-enabled devices
  - TBox (ontology) fragmented in chunks scattered across network nodes
  - ABox (individuals) physically associated to distinct devices
- Different u-KBs can coexist in the same domain (via unique ontology URIs)
- On-demand retrieval of just the ontology chunks needed for reasoning

Ubiquitous Knowledge Base (2/2)



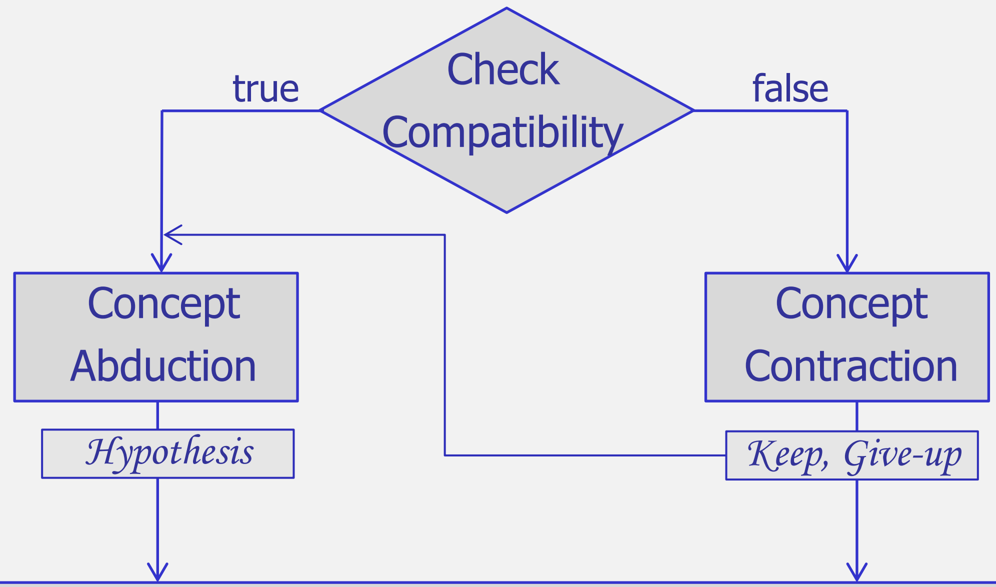
- Enable ontology dissemination and on-the fly rebuilding
- Class hierarchy translated to a nested numbered list:
  - each class has a unique associated ID
  - subclasses introduce an additional dot-separated number
  - sibling classes IDs differ by the last incremented number
- The ontology includes IDs as OWL class annotation properties
- Upper Ontology chunk shared by all nodes, with the topmost levels in the class hierarchy

Service/resource discovery

Input

- Request: logic-based annotation of requested service/resource
- Resource: logic-based annotation of service/resource

Semantic Matchmaking



Penalty (semantic distance)

$$d(S,R) = 100\ [\ 1 - (\ penalty_c(S,R) + penalty_a(S,R)\ )\ /\ max\_penalty_a(R)\ ]$$

Output

Ranked list of services/resources

Case Study: resource discovery in a semantic-based swarm

A surveillance mission requires environmental temperature monitoring. The system consists of five nodes connected in a swarm-based sensor network:

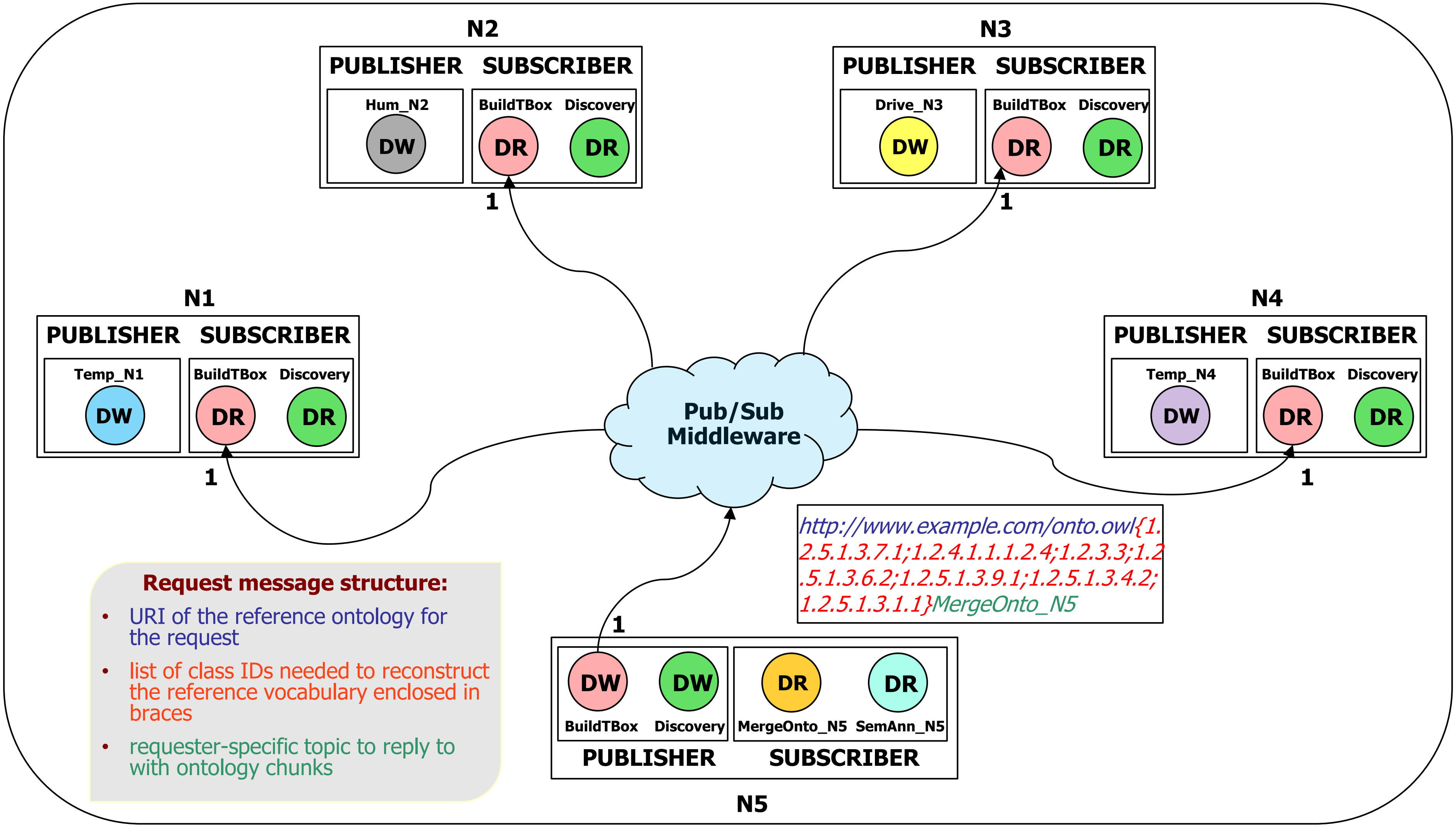
- N1: temperature sensor
- N2: humidity sensor
- N3: floodlight drive
- N4: temperature sensor

→ Resource/service providers

- N5: mission controller → Service requester

- The system exploits a publish/subscribe middleware for communication
- Each node includes:
  - Publisher for data dissemination through one or more Data Writer (DW) objects
    - allowing data to be published under a given Topic
  - Subscriber for data gathering through one or more Data Reader (DR) objects
    - each associated to one Topic subscription

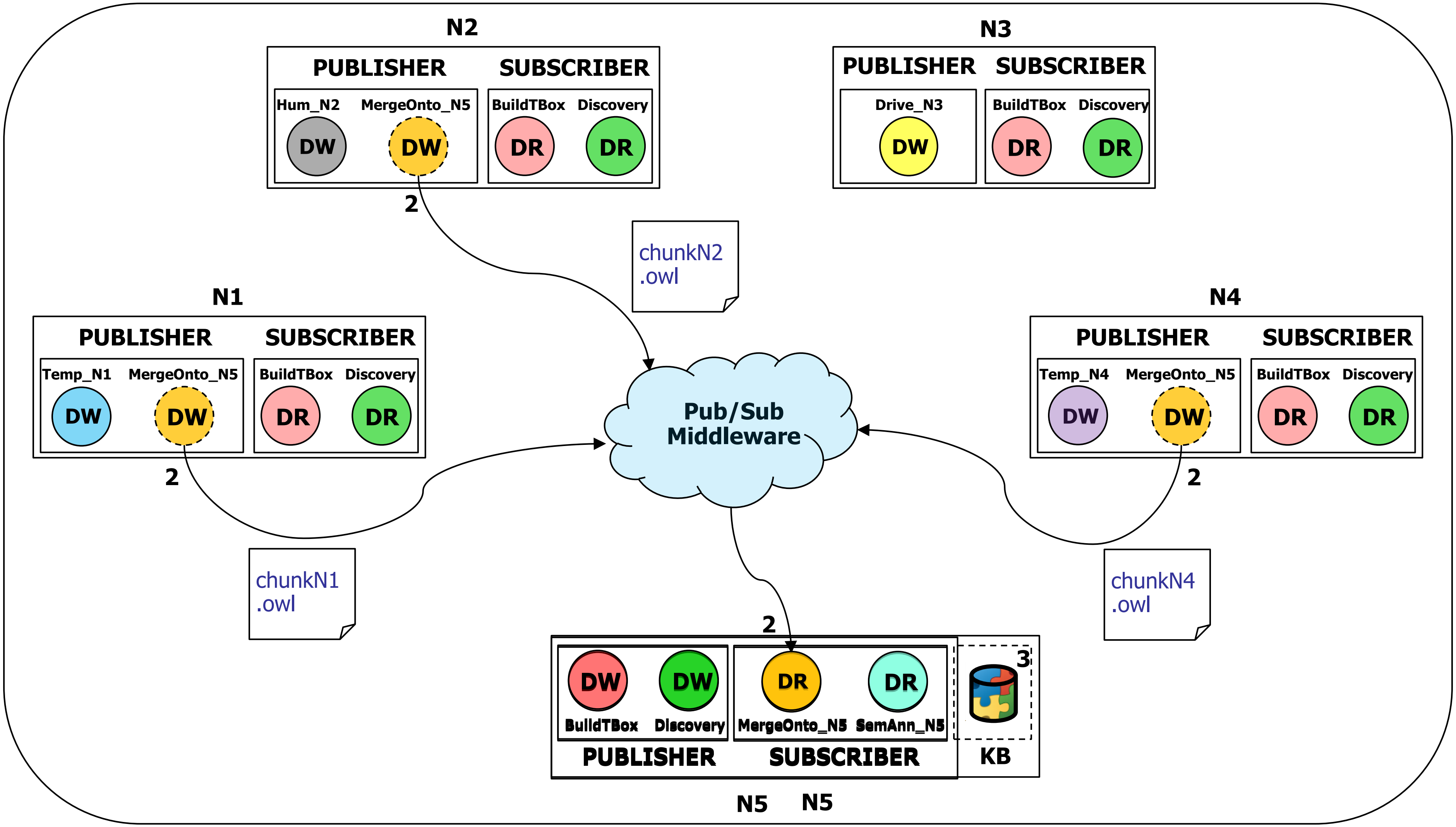
Case Study: ontology rebuilding (1/2)



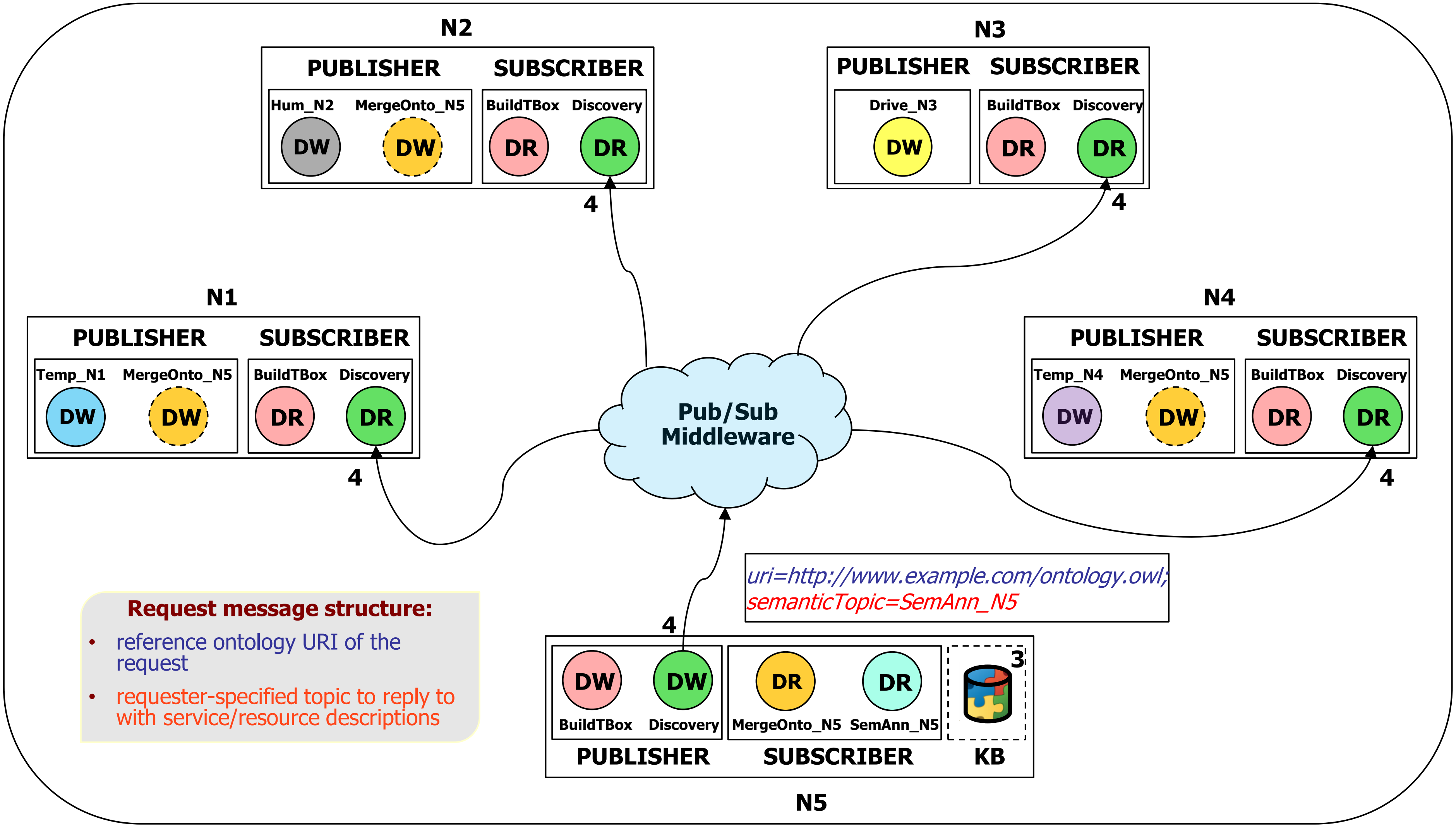
Request message structure:

- URI of the reference ontology for the request
- list of class IDs needed to reconstruct the reference vocabulary enclosed in braces
- requester-specific topic to reply to with ontology chunks

Case Study: ontology rebuilding (2/2)



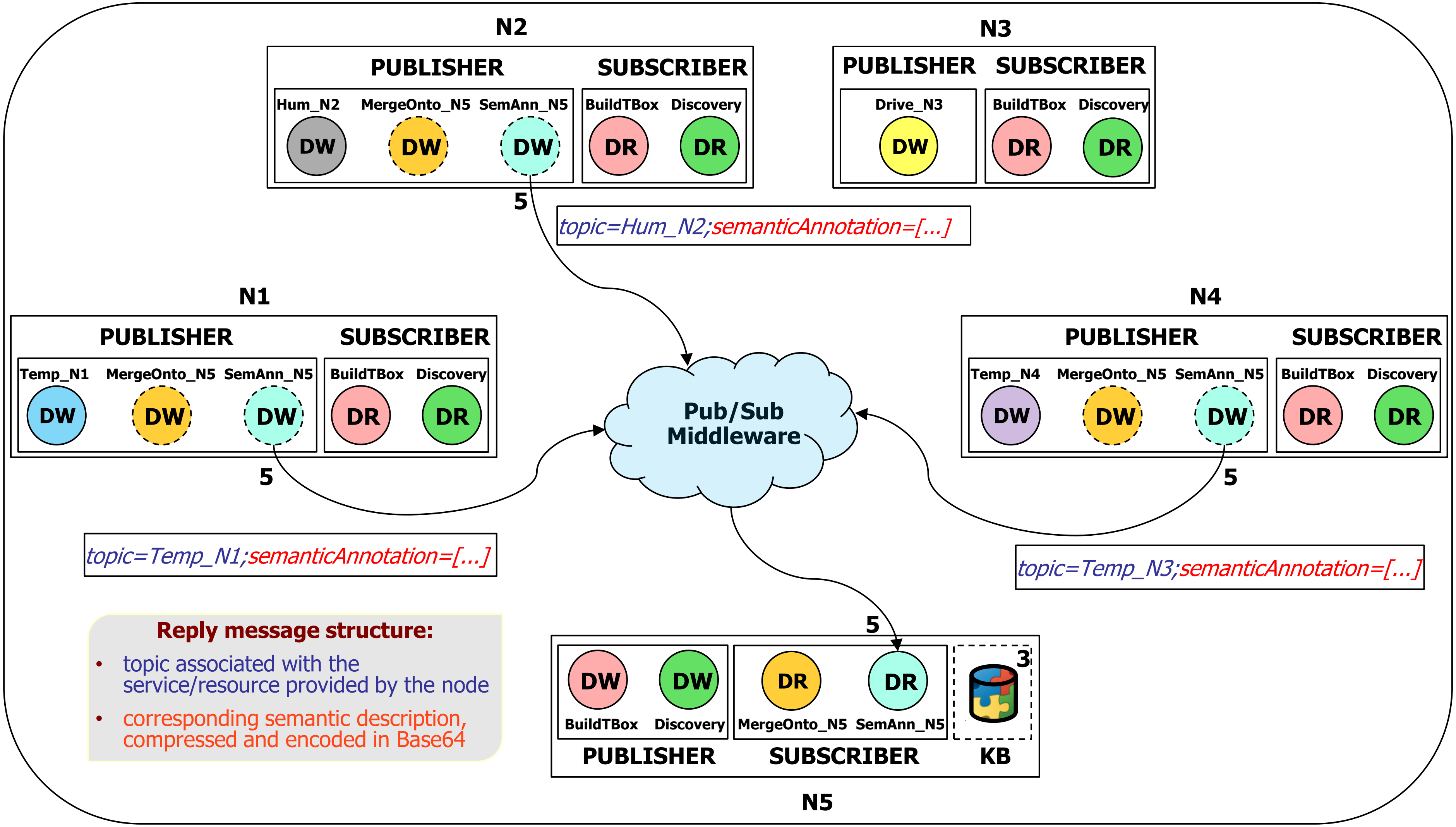
Case Study: resource allotment (1/3)



Request message structure:

- reference ontology URI of the request
- requester-specified topic to reply to with service/resource descriptions

Case Study: resource allotment (2/3)

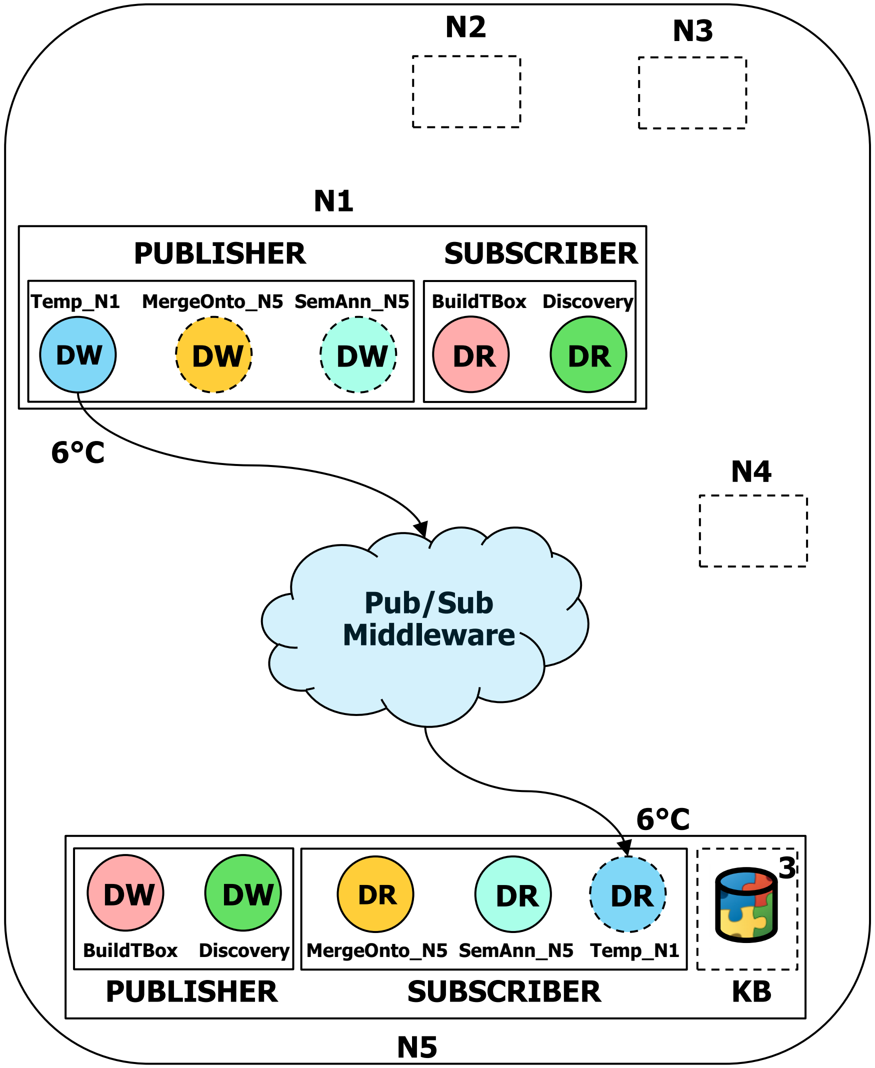


Reply message structure:

- topic associated with the service/resource provided by the node
- corresponding semantic description, compressed and encoded in Base64

Case Study: resource allotment (3/3)

Semantic Description	Score
N5 (request): TemperatureSensor ⊓ ∀observes.Temperature ⊓ ∀hasMeasurementProperty. (HighAccuracy ⊓ LowMeasurementRange ⊓ LowFrequency ⊓ HighPrecision ⊓ HighResponseTime)	N.A.
N1: TemperatureSensor ⊓ ∀observes.Temperature ⊓ ∀hasMeasurementProperty. (HighAccuracy ⊓ LowFrequency ⊓ MediumMeasurementRange ⊓ HighPrecision ⊓ MediumResponseTime ⊓ MediumResolution ⊓ LowLatency)	**92,6**
N4: TemperatureSensor ⊓ ∀observes.Temperature ⊓ ∀hasMeasurementPropert. (LowAccuracy ⊓ LowFrequency ⊓ LowMeasurementRange ⊓ LowPrecision ⊓ MediumResponseTime ⊓ LowResolution ⊓ LowLatency)	85,2
N2: HumiditySensor ⊓ ∀observes:Humidity ⊓ ∀hasMeasurementProperty.(LowAccuracy ⊓ LowFrequency ⊓ LowMeasurementRange ⊓ MediumPrecision ⊓ MediumResponseTime ⊓ LowResolution ⊓ LowLatency)	71,4



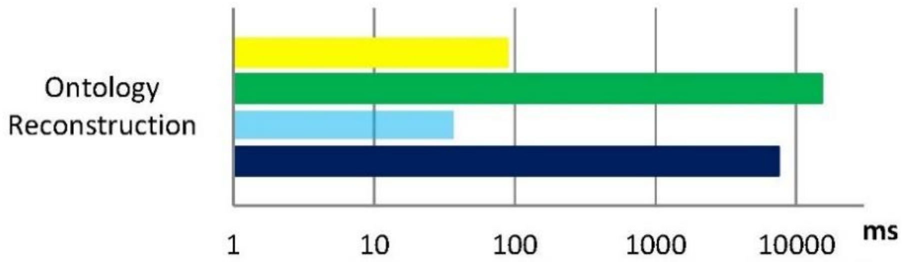
Implementation

- **Java** language
- **Bee-DDS** (Bee-Data Distribution Service) platform

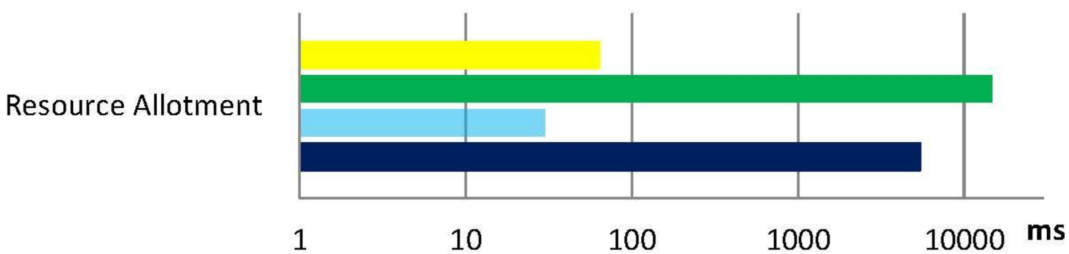
Performance evaluation

- Realistic stress test: 500 resource/service providers, 10 requesters
- Requesters deployed on 3 VMs
  - dual-core CPU, 2 GB RAM, 32-bit Ubuntu 14.04 LTS, 32-bit Java 8 SE
- Providers equally distributed among 50 VMs
  - dual-core CPU, 800 MB RAM, same operating system and Java runtime environment
- Performance metrics:
  - Turnaround time
  - RAM usage

Experimental results: turnaround time



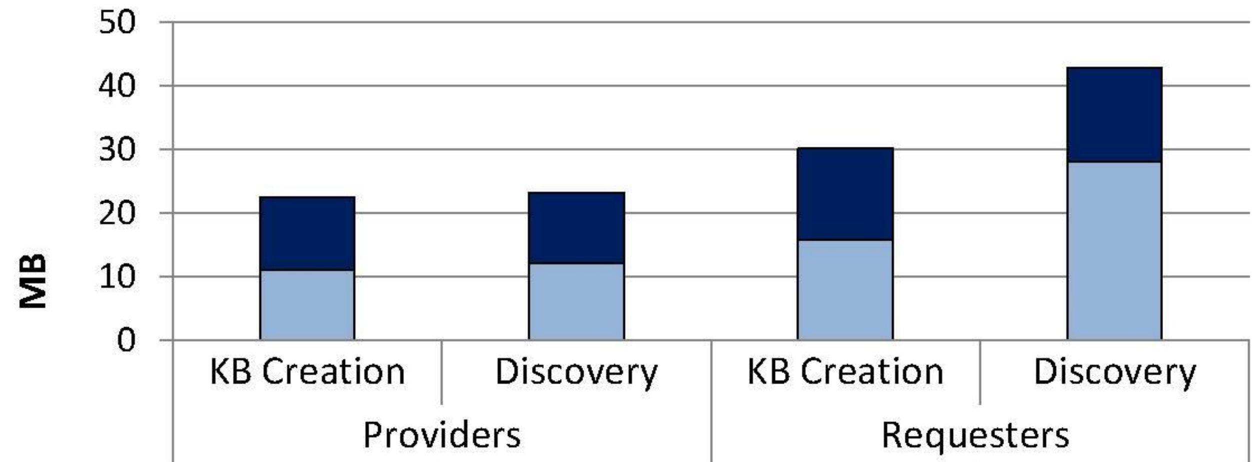
	Ontology Reconstruction
Compression	87.84
Transmission	15066.79
Decompression	36.78
Chunk Merging	7471.45



	Resource Allotment
Compression	63.18
Transmission	14714.92
Decompression	29.88
Semantic Matchmaking	5397.01

- Information transmission has the highest impact
- Relatively higher turnaround time than basic swarm middleware
  - Worthwhile in advanced scenarios where managing complex information is required

Experimental results: memory usage



	Providers – KB Creation	Providers – Discovery	Requesters – KB Creation	Requesters – Discovery
Non-heap Memory	11.36	11.07	14.42	14.79
Heap Memory	11.12	12.17	15.74	28.05

- Acceptable memory usage for current single-board computers and embedded devices (*e.g.* Raspberry Pi)

Conclusion and Future work

Main contribution

- A layered architecture for semantic-based service/resource discovery and allocation in swarm-centric sensor and actor networks
- Off-the shelf publish/subscribe middleware (*Bee-DDS*) for inter-node communication
- Logic-based structured annotation of available services and requests
- Dynamic discovery via deductive matchmaking and results ranking by semantic relevance
- Implemented working prototype to prove both correctness and feasibility of the proposal in a practical case study

Future Work

- Performance optimizations
- Further semantic-based SOA features, including service composition, clustering, substitution and requester/provider negotiation