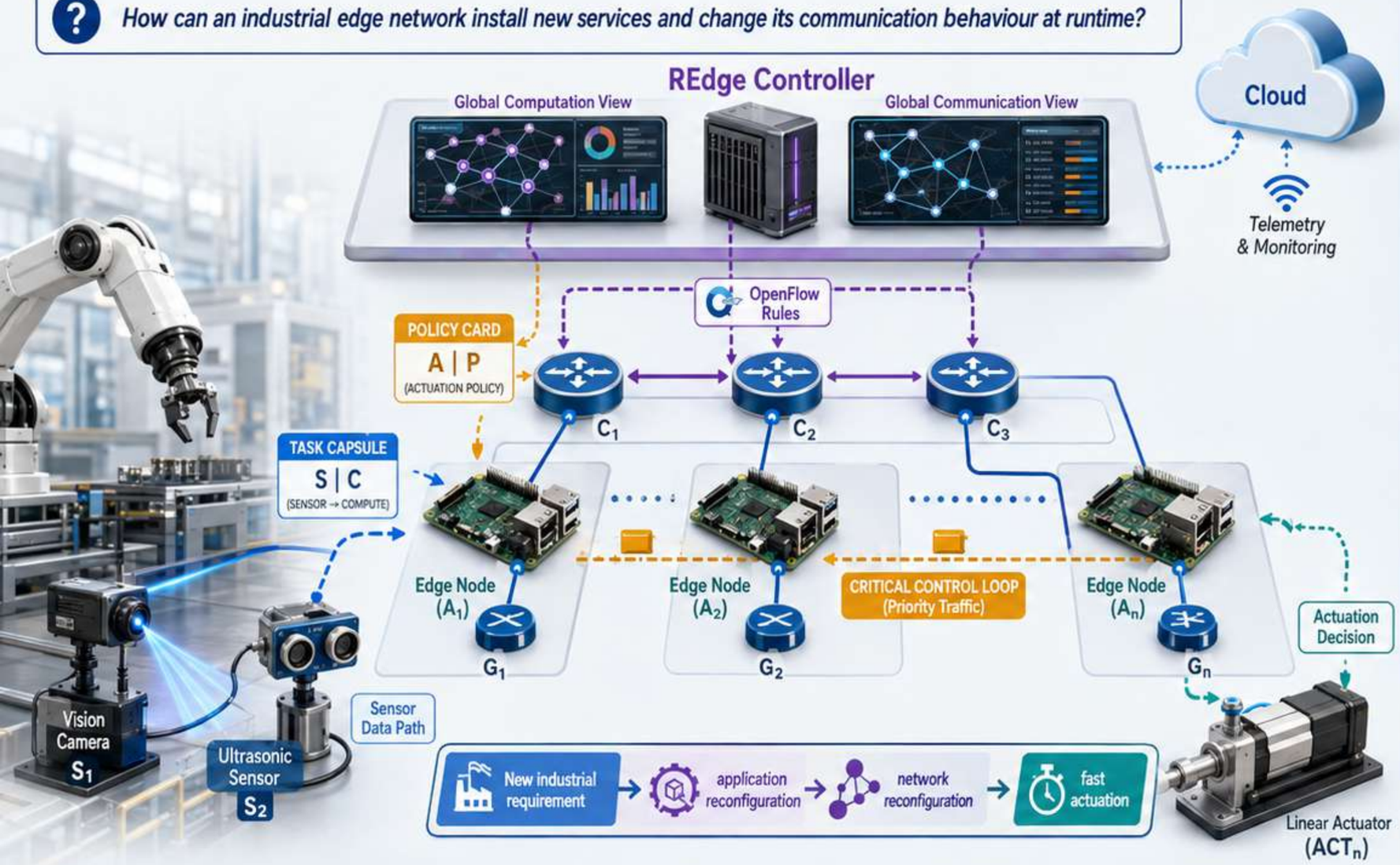
REdge: Programmable Industrial Intelligence at the Edge

SDN-based reconfiguration of application services and network behaviour

? *How can an industrial edge network install new services and change its communication behaviour at runtime?*



HIGHLIGHTS OF THE PAPER

01 JOINT APPLICATION AND NETWORK RECONFIGURATION

REdge remotely changes application modules, actuation policies and communication parameters at programmable edge nodes.

02 GLOBAL COMPUTATION AND COMMUNICATION VIEWS

The SDN-enabled controller observes edge resources and network paths to coordinate task execution, offloading and forwarding.

03 QOS-AWARE CONTROL-LOOP FORWARDING

Priority-aware OpenFlow placement supports sensor-to-actuator traffic when the devices belong to different edge networks.

04 EXPERIMENTAL FINDINGS

Testbed and simulation results show lower actuation time, network usage, energy consumption, execution time and resource cost than conventional edge and cloud architectures.

Reported improvement: up to **200%** *vs edge baseline and* **400%** *vs cloud baseline in actuation time discussion.*

Validated using Raspberry Pi edge nodes, ESP8266 motes, sensors, actuators, OpenFlow and an SDN core.

Reconfigure the service and the network together instead of replacing industrial hardware.

IEEE Internet of Things Journal, 2023 • DOI: 10.1109/JIOT.2023.3268375

When an Industrial Task Changes, the Network Must Change Too

Dynamic sensor-to-actuator control cannot rely on fixed edge services

1 INDUSTRIAL EVENT

Warehouse robot arm moving toward a restricted zone.

Sensors detect threshold crossing; alert requirement.

TASK CAPSULE

S | C

S: input and program size

C: required CPU cycles

task request (S_k, C)

$C = S_k \times D_i$

2 FIXED EDGE LIMITATION

Edge node has only a limited installed task set.

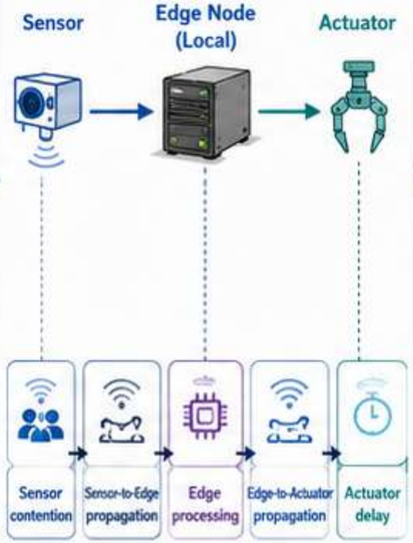
New task or changed actuation threshold requested.

- module unavailable
- manual reprogramming
- delay
- cloud dependency
- inconsistent network configuration

Fixed edge services and device-by-device configuration are slow, unscalable and prone to error.

3 SAME-NETWORK CONTROL LOOP

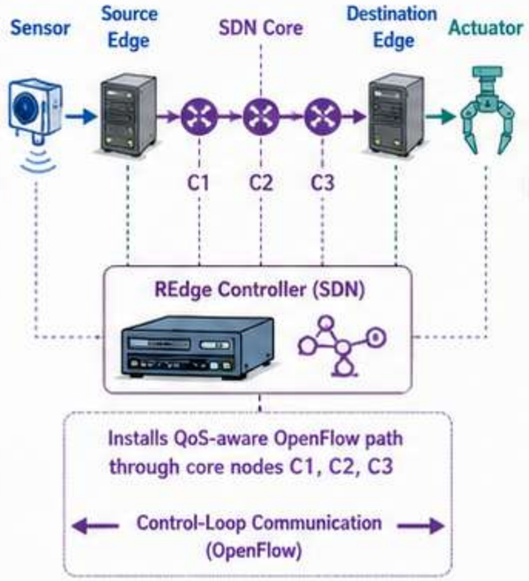
Same edge network



Local processing enables immediate actuation.

4 DIFFERENT-NETWORK CONTROL LOOP

Different edge networks



Core-network delay and forwarding policy now matter.

5 REDGE RESPONSE

- Install or modify Application Module
- Change Actuation Policy
- Set task priority
- Install OpenFlow path (Priority Flow Placement)
- Select target Edge Node
- Trigger actuator

POLICY CARD

A | P

A: Actuation Policy **P:** Priority

REdge coordinates computation and communication for the complete control loop.

RESEARCH GAP | Existing approaches usually reconfigure either services or networking. REdge handles both in one architecture.

RESEARCH TAKEAWAY | Industrial reliability depends on changing the task and its network path together.

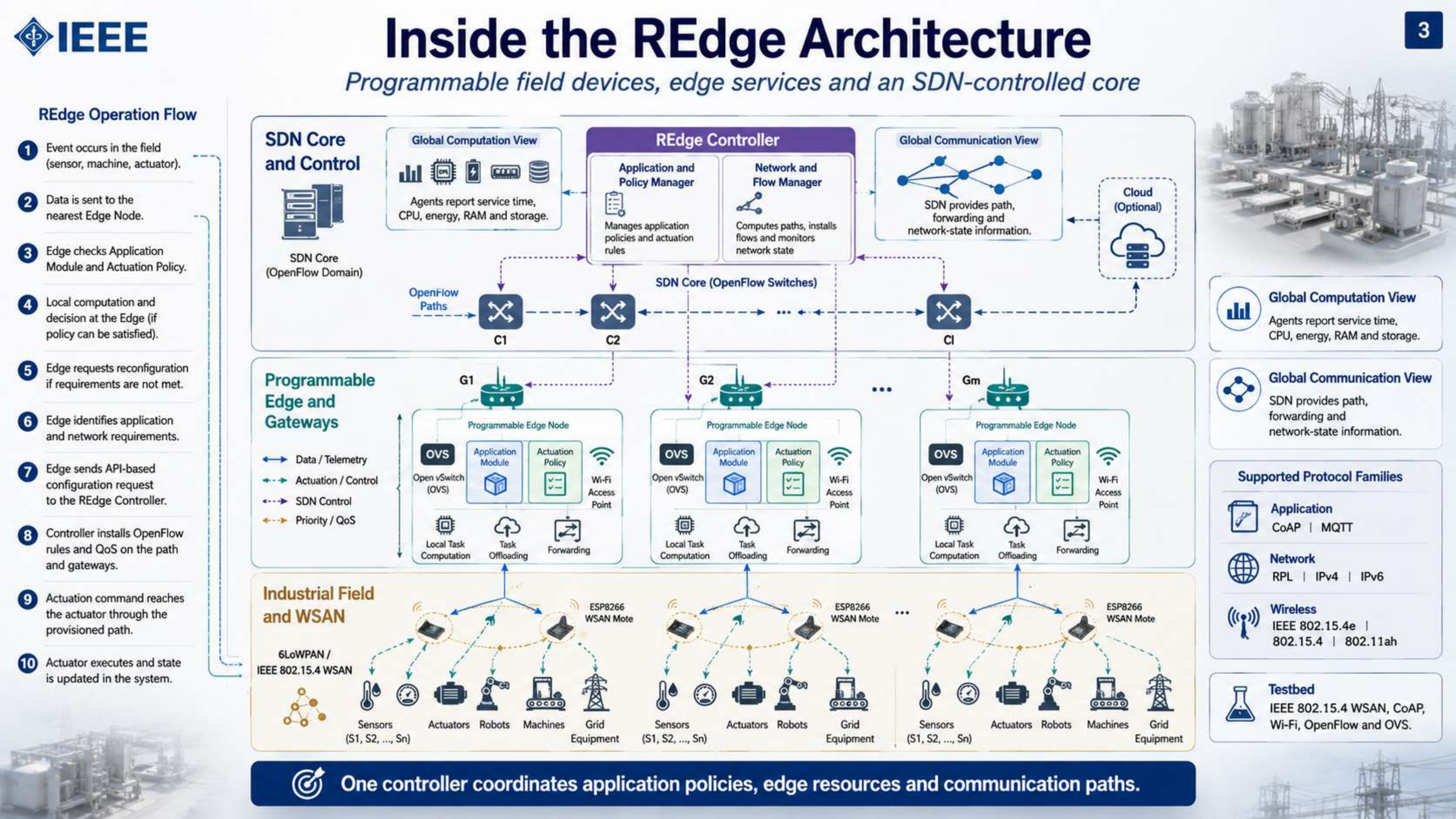
IIoT: Industrial Internet of Things | WSAN: Wireless Sensor and Actuator Network

Inside the REdge Architecture

Programmable field devices, edge services and an SDN-controlled core

REdge Operation Flow

1. Event occurs in the field (sensor, machine, actuator).
2. Data is sent to the nearest Edge Node.
3. Edge checks Application Module and Actuation Policy.
4. Local computation and decision at the Edge (if policy can be satisfied).
5. Edge requests reconfiguration if requirements are not met.
6. Edge identifies application and network requirements.
7. Edge sends API-based configuration request to the REdge Controller.
8. Controller installs OpenFlow rules and QoS on the path and gateways.
9. Actuation command reaches the actuator through the provisioned path.
10. Actuator executes and state is updated in the system.



One controller coordinates application policies, edge resources and communication paths.

Global Computation View

Agents report service time, CPU, energy, RAM and storage.

Global Communication View

SDN provides path, forwarding and network-state information.

Supported Protocol Families

Application
CoAP | MQTT

Network
RPL | IPv4 | IPv6

Wireless
IEEE 802.15.4e | 802.15.4 | 802.11ah

Testbed

IEEE 802.15.4 WSAN, CoAP, Wi-Fi, OpenFlow and OVS.

How REdge Reprograms Tasks and Prioritises Control Traffic

Priority, fairness and QoS-aware forwarding at runtime

Dynamic Task Reconfiguration

Programming Requests → REdge Controller

Each request contains:
- Module (Application Module)
- Policy (Actuation Policy)
- Programmable parameters *P*
- Priority *p*

1. Receive programming requests.
2. Identify affected Application Module.
3. Read module-policy priority.
4. Send programming message to Edge Node.
5. Receive acknowledgement.

Positive ACK	Negative ACK	No Response
Create API with selected *P*.	Increase priority to $p + 1$.	Wait τ.
Program highest-priority module.	Place request in fair retry queue.	Resend request.
Update Actuation Policy.		

Failed requests gain priority, preventing permanent starvation.

S | C — TASK CAPSULE (S | C)

A | P — POLICY CARD (A | P)

Application-to-Network Interface

- Application Type (Critical / Noncritical)
- Policy Translation (Actuation Policy)
- Flow Priority and Rules (Priority *p*, flow rules)
- Target Edge / Actuator (Edge node or device)

The current interface mainly translates application needs into priority and flow-rule changes.

REdge Controller (SDN Brain)

Modifies Flow Table
- Match (Flow)
- Priority
- Actions
- Timeout

Receives Packet-In Metadata
- Resource Name
- Resource Version
- Destination Path
- Flow / Packet Info

Global Resource State

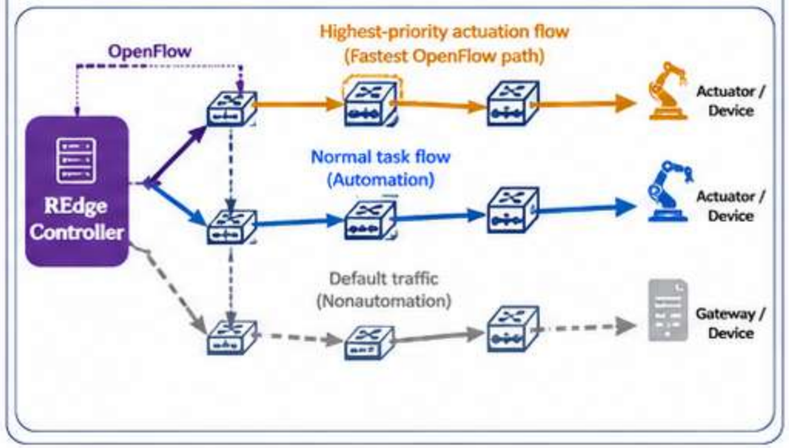
Service | CPU | Energy | RAM | Storage

Global state enables constant-time target-edge selection in the proposed model.

Priority Flow Placement

Critical Automation (Actuation) | Ordinary Automation (Tasks) | Monitoring Data | Background Traffic

1. Classify flows by resource use and priority.
2. Detect automation-related packets.
3. Compare priority values.
4. Send highest-priority flow first.
5. Use first-come, first-served when priority is equal.
6. Forward nonautomation traffic using default rules.

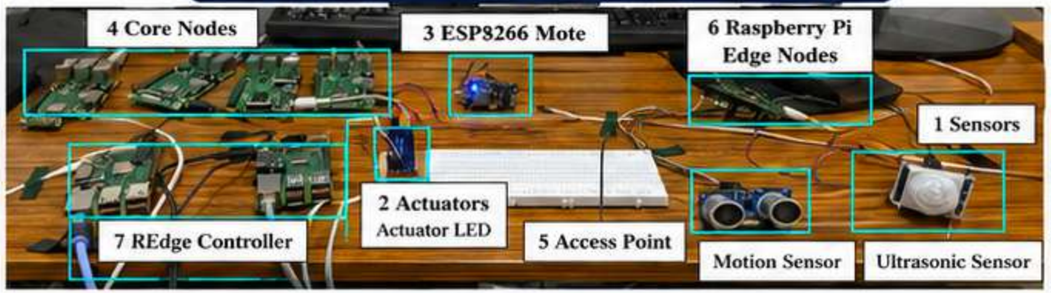


Program the most important service first and give its control traffic the right network priority.

What the REdge Experiments Show

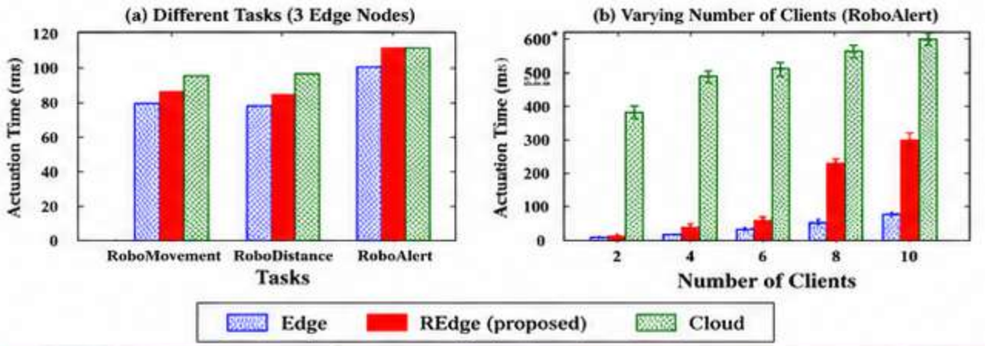
Testbed and simulation comparison with conventional edge and cloud architectures

TESTBED: PROPOSED ARCHITECTURE (FIG. 4)



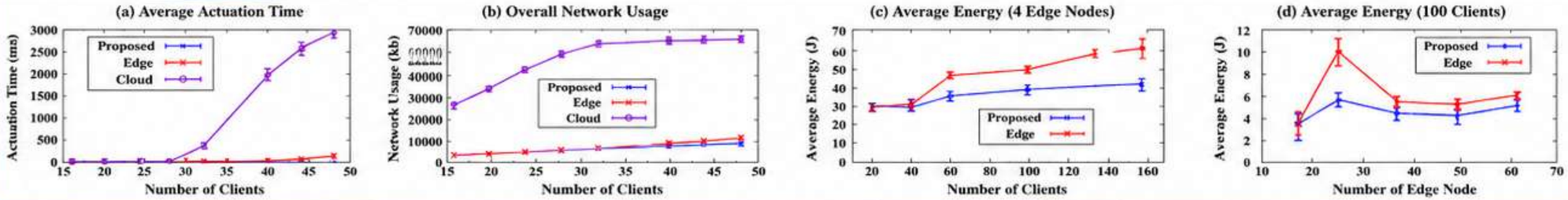
Tasks: *RoboMovement, RoboDistance, RoboAlert*

RESULT AREA 1: ACTUATION TIME (FIG. 5)



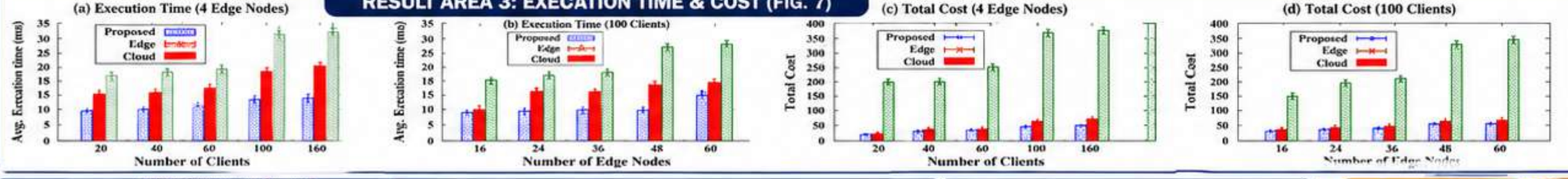
Dynamic task reconfiguration and SDN forwarding reduce sensing-to-actuation delay.

The paper reports up to 200% and 400% actuation-time improvement relative to its edge and cloud baselines.



Runtime policy updates reduce waiting, communication and the number of edge nodes required for a task.

RESULT AREA 3: EXECATION TIME & COST (FIG. 7)



The programmable architecture completes more tasks with fewer edge resources and incurs the lowest cost among the compared approaches.

EXPERIMENTAL SETUP

Raspberry Pi Edge Nodes · ESP8266 Motes · Motion Sensor · Ultrasonic Sensor · Actuator

IEEE 802.15.4 WSAN · CoAP · Wi-Fi Access Point · OpenFlow · Open vSwitch

20	Up to 60	Up to 160
Core Nodes in Simulation	Edge Nodes in one Simulation Study	Clients in Plotted Experiments

CURRENT SCOPE

The implemented application-network interface mainly supports priority and flow rules. Fault handling, security, jitter and broader heterogeneous-application mapping remain future extensions.

CONCLUSION CHAIN

Programmable services → Dynamic flow rules → Priority-aware control loop → Faster actuation → Lower network, energy and resource cost

FINAL RESEARCH CONCLUSION

REdge jointly reconfigures industrial application policies and network behaviour, improving task availability and reducing actuation and offloading delay.

FUTURE WORK

- Fault and failure handling for edge devices
- Learning-based flow classification
- Learning resource availability
- Improved QoS-aware decision making

Reprogram the service. Reconfigure the path. Actuate faster.