



daas

Device as a Service

Technology overview: state of the art, positioning, practical value, application sectors and measurements

Paolo Lo Giacco — p.logiacco@sebyone.it
Sebastiano Meduri — s.meduri@sebyone.it
Lorenzo Grillo — l.grillo@sebyone.it
Matteo Pagano — m.pagano@sebyone.it

3 November 2026 · libdaas 0.24.4 · daas-protocol 6.1.0



Sebyone Srl · Via Pola 8, 87100 Cosenza · Piazza Capranica, Rome · Italy
sebyone.it · daasiot.com · info@sebyone.it · developers@sebyone.it
daas-api (libdaas) is released under the MPL-2.0 licence · © 2019–2026 Sebyone Srl

Foreword

This document builds on and updates the official presentation of daas at the IEEE workshop, “Sebyone: Connecting the Future of IoT”, which introduced the Device as a Service Overlay Mesh paradigm and its four core capabilities: an overlay network independent of the physical media, protocol channelling, self-synchronisation (dATS) and self-provisioning. Compared with that presentation, this document adds the state of the art, the positioning, the results of the papers on the identity and routing model, and the measurements taken on version 0.24.4 of the library.

What daas is

daas (Device as a Service) is a networking technology that gives every device an identity of its own, independent of its physical address, and lets it talk to the others over any transmission medium, with no server, no configuration and no network infrastructure.

The problem it solves is easy to state. Today a device exists on a network only if someone assigns it an address, registers it on a server, tells it how to reach the others and gives it a common clock. Each step is an installation cost and a point of failure: if the server, the DHCP or the Internet link goes down, devices stop talking to each other even when they are a metre apart.

With daas the network builds itself. A device switched on near another discovers it, synchronises with its time and receives an identity. From that moment it can exchange data with any member of the network, across several hops and across different physical media (Ethernet, Wi-Fi, Bluetooth, LoRa, Zigbee, serial). The identity follows the device: if it changes IP address, radio channel or path, it remains the same node.

The reference library is daas-api (libdaas), written in C++, released under the MPL-2.0 licence and available on Linux, macOS, Windows, Android, iOS, Raspberry Pi, OpenWrt and ESP32. daas-iot is the Sebyone product built on top of it. The official site is daasiot.com.

State of the art: how devices connect today

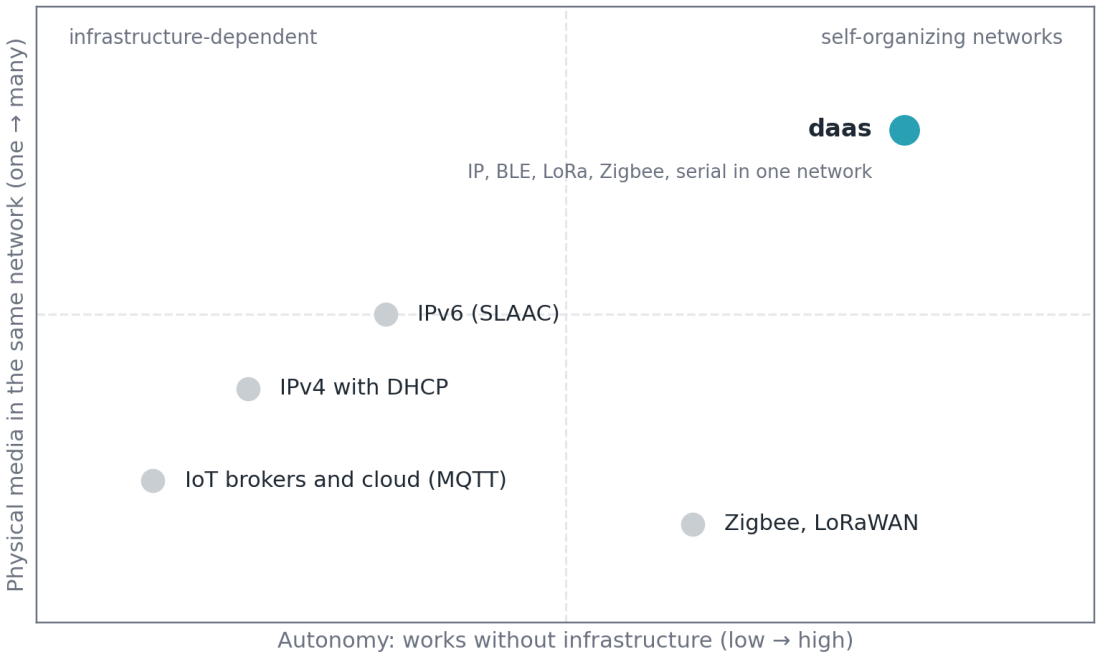
Every mature solution entrusts device identity to something external to the device itself: a registry, a server or a pre-assigned chip. None carries in the address the moment and the place at which the node joined the network.

Approach	Who assigns the identity	What it needs to work	Main limitation
Classic IP (IPv4, DHCP)	a DHCP server or an administrator	an address plan and a reachable server	the address is a locator, not an identity: it changes when the node moves
IPv6 (SLAAC, RFC 7217)	the node itself, from the MAC or a hash	a router advertising the prefix, plus Duplicate Address Detection	zero-touch only with infrastructure; the address says neither when nor where the node was born
IoT brokers and cloud (MQTT, cloud platforms)	the platform, with per-device credentials	continuous Internet connectivity to the broker	without Internet, neighbouring devices cannot talk; recurring per-node costs
Low-power IoT networks (Zigbee, LoRaWAN)	the coordinator or the network server, at join	a gateway or coordinator per segment	one physical medium per network; the address is allocated by a central entity
Time-seeded identifiers (UUID v1/v7, Snowflake)	a unique generator plus a counter	a generator identifier assigned in advance	they solve uniqueness, not connectivity: they are not a network

The closest precedent to daas is IPv6: network prefix plus node identifier, hash-based auto-generation, duplicate detection. The notion of rank, the hop distance from the root, already exists in RPL (RFC 6550) for low-power networks. daas takes these ideas and brings them together in a single model, placing inside the identity what the others keep out: the time of birth and the position in the topology.

Where daas stands

daas occupies the space the others leave uncovered: networks that must exist and organise themselves without infrastructure, across different physical media. IPv6 optimises interoperability with the rest of the world; daas optimises the autonomy of the segment.



Positioning map, qualitative: further right means less infrastructure needed; higher means more physical media coexisting in the same network.

The two models are complementary, not alternatives. daas can operate as an identity and self-organisation layer beneath or beside IP: the segment forms, synchronises and assigns identities on its own; an IP gateway gives it, when needed, reachability from the Internet. Compared with cloud brokers, daas needs neither continuous connectivity nor a per-device fee; compared with Zigbee and LoRaWAN, it is not tied to a single physical medium and does not depend on a coordinator.

In short, daas is the choice when:

- devices must talk to each other even without Internet or servers;
- the network changes often (units joining, leaving, being replaced);
- different physical media coexist (IP, Bluetooth, LoRa, Zigbee, serial);
- the installation cost per unit must tend to zero.

The most innovative aspect for research

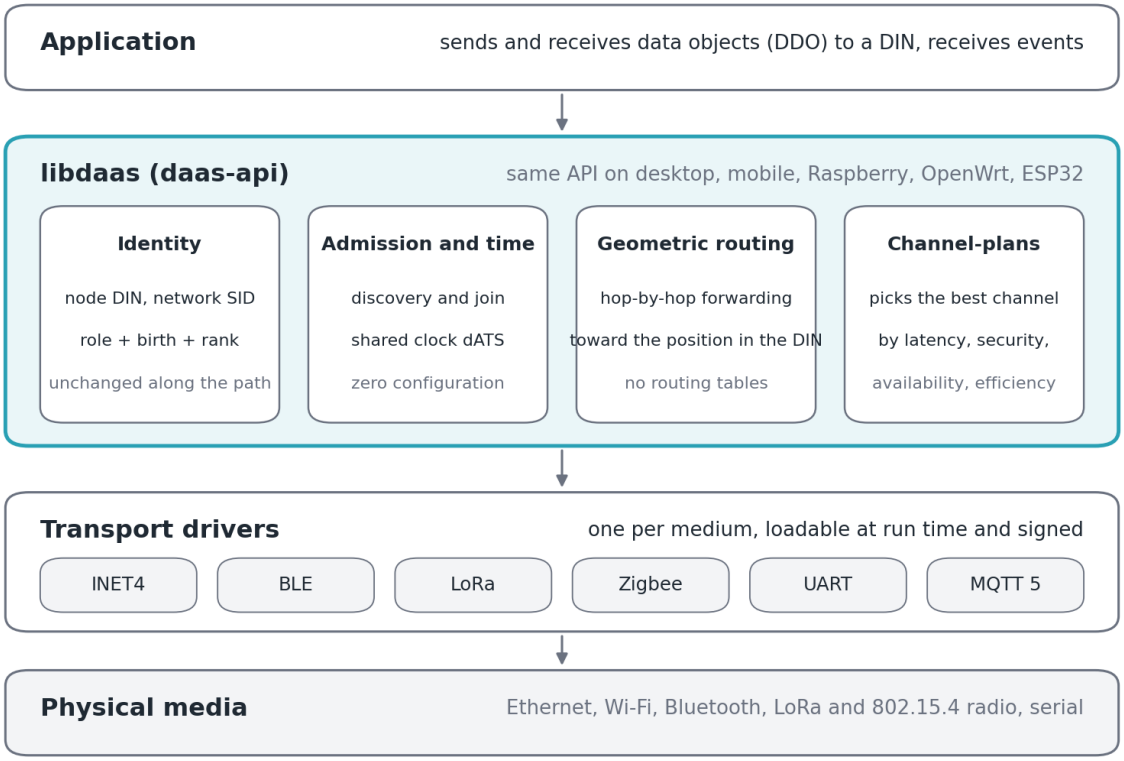
From the standpoint of networking research, the novelty of daas is neither a faster protocol nor a new physical medium: it is the idea that a node's identity is a coordinate. In every existing architecture the address is an administered locator, assigned by a registry or a server, and it says nothing about who the node is, when it joined and where it sits in the topology. In daas the identity embeds the time of birth and the position in the admission tree, and from this follow three properties that elsewhere require three separate systems: uniqueness without a registry, because the identity is unique by construction; routing without tables, because the destination is computed from the identity itself; verifiability without cryptography, because an identity inconsistent with time and position can be recognised by anyone. To this is added the founding condition of the model: a node exists only when it has an active channel and a shared time with another node. The network is not configured; it emerges.

Why this research matters, with respect to other groups working on the same themes. The themes daas brings together are studied today, separately, by distinct and active communities. The IETF has worked for years on the self-organisation of low-power networks (RPL and 6LoWPAN in the ROLL and 6lo groups), on IPv6 address auto-generation and its cryptographic binding (RFC 7217, CGA), and with the ANIMA group on networks that bootstrap and manage themselves. Academic research on geometric and greedy routing, from the hyperbolic embeddings of the Internet (Krioukov, Papadopoulos, Boguñá) to Kleinberg's work on table-free routing, has shown that a coordinate can replace routing tables, but did so by assigning coordinates to existing networks, not by letting them be born with the identity. Peer-to-peer networks and the decentralised Web (Kademlia, libp2p, IPFS, the W3C Decentralized Identifiers) have made identity independent of the server, but anchor it to cryptographic keys and to an underlying IP network. Industry alliances (Thread, Matter, LoRa Alliance) have made device onboarding simple, but within a single physical medium and with a coordinator. daas sits at the point where these lines do not yet meet: an identity born with the network, carrying time and position, serving as a coordinate for routing and verifiable before cryptography, over any physical medium. It is a direction the community is converging on from different sides, and this is why the daas results, in particular uniqueness by construction and identity-computed routing, are relevant beyond the project: they can be compared, verified and taken up by these groups, while their mature results, regeneration on collision and cryptographic binding, point daas to its next steps.

Technical-scientific committee and university partnerships. Research on daas is accompanied by a technical-scientific committee that steers its development lines, evaluates its results and checks their consistency with the state of the art. The project also maintains collaboration with two universities, sharing research activities, theses and experiments on the identity model, on time synchronisation and on geometric routing: an exchange that guarantees the scientific rigour of the results and opens the way to publications and independent validation.

How it works, in plain words

A daas node becomes one the moment it opens a channel with another node and synchronises with its time. From then on four mechanisms do the rest.



Layered architecture: the application sees neither IP addresses nor physical media; it talks to a DIN and the library does the rest.

Identity (DIN and SID). Every node has a DIN, a 64-bit number that identifies it. Every network has a SID, which identifies the segment. The DIN is made of a part chosen by the application (the role: sensor 12, arm 3), a temporal part assigned at the moment of joining and a rank, the hop distance from the node that founded the network. The DIN does not change if the node changes address, medium or path.

Admission (discovery and join). A new node sends a search signal. A node of the network answers, synchronises it and assigns it a DIN and a position. No operator intervenes. The network grows as a tree: every node enters through another, and its position in the tree ends up inside its identity.

Shared time (dATS). All nodes of a network share the same clock, with a configurable maximum error (100 ms by default). A node sends application data only once it is synchronised: data written earlier stays queued and leaves as soon as the node is ready.

Data exchange and routing. The application sends data objects (DDO) to a DIN; the library picks the best channel among those available (by latency, security, availability or efficiency, according to the channel-plans) and, if the recipient is not nearby, forwards hop by hop toward its position in the tree. Every intermediate node forwards without knowing the recipient: this is geometric routing. Transport is a detail of the driver: the same node can have IP, Bluetooth, LoRa and serial channels toward different neighbours.

On top of this there is a last-resort defence: under attack, any node can isolate the entire reachable network (attack panic), and every node stays down until an operator restarts it.

Addressable space and routing on the complex plane. The address space is wide enough never to be a constraint: each segment can host up to 2^{64} nodes and a federation of segments up to 2^{128}, without anyone

keeping a registry of individual nodes, because uniqueness is guaranteed by the way the identity is built at admission. The evolving routing model also gives this identity a geometric reading: the DIN is interpreted as a point on a complex plane, the Gauss plane, normalised with respect to the node that founded the network. The time of birth and the position in the tree become the two coordinates of the point, and the network reads as a map in which every node knows, from the recipient's identity alone, in which direction and in which sector to forward. The path is neither searched for nor stored: it is computed. This makes routing deterministic, light for the smallest devices and independent of the number of nodes, and the same principle can be extended to federated networks. The mathematical formalisation and the implementation choices are described in the dedicated papers.

Practical value

The fundamental advantage is one: uniqueness and the organisation of the network move from the administrative plane to the protocol plane. What today requires an operator, an address sheet or a central service becomes a property of the network itself. An identity unique by construction, dated at birth and positioned in the graph, is more than an address: it is a verifiable assertion about who the node is, when it joined and where it stands.

Problem today	What changes with daas
address plan and manual registration of every unit	no plan and no registry: the provisioning cost per unit tends to zero
dependence on a registry or a server reachable online	offline operation: allocation is local to the segment
reconciliation between network identity and logical asset (time series, digital twin)	the pair (user part, temporal part) is already (asset, incarnation): no external map
unreliable clocks, non-unique event ids	shared time and ordered identities: globally unique and ordered events
duplicate addresses after two partitions rejoin	transient, self-correcting conflict, detected by construction
exhausted or fragmented address space	2^{64} nodes per segment, with no registry of individual nodes

On the security side, the daas identity offers three zero-cost plausibility checks, before and independently of any cryptographic layer: a DIN whose temporal part predates the network or lies in the future is forged; a node whose rank is inconsistent with the node that admitted it is anomalous; the same DIN appearing from two distinct addresses signals a cloned device. These checks raise the cost of an attack and give cryptography a solid base to anchor to, but they do not replace authentication of network admission, which today is left to the application layer.

For developers the cost is contained: one C++ library with the same API on every platform, two execution modes (with its own threads on desktops, servers and smartphones; cooperative loop on microcontrollers) and a few bytes of state per node on embedded platforms.

Application sectors

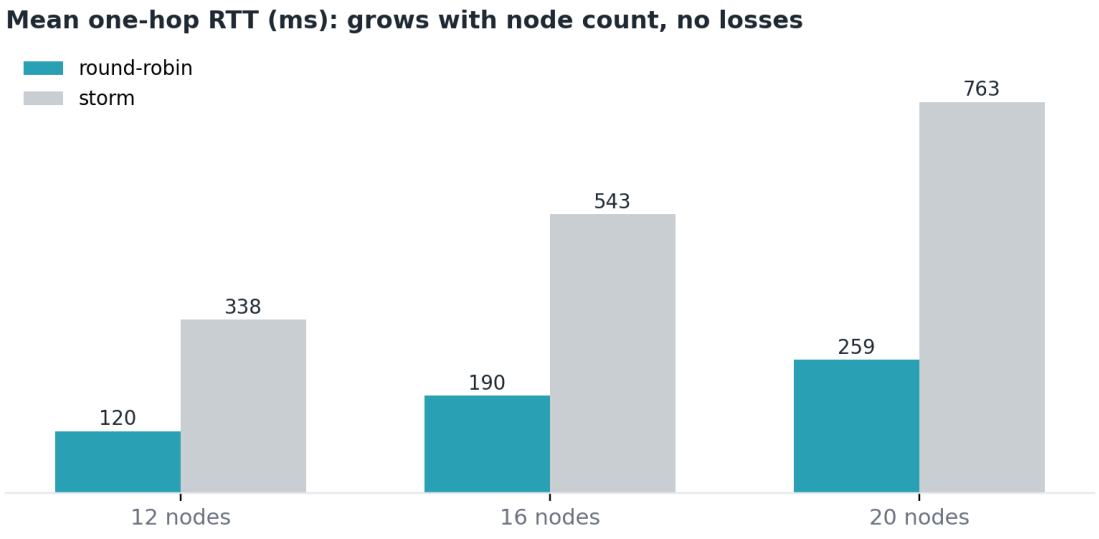
daas is useful wherever up-front configuration is impractical: units entering and leaving radio range, field replacements, sites without infrastructure.

Sector	Use case	What daas brings
Robotics and swarms	fleets of robots, AGVs and drones that form and regroup in the field	joining a formation with no operator; the rank in the identity gives the distance from the leader or base and helps choose relays; a replaced robot inherits the role and receives a new incarnation, so logs stay consistent
Industrial IoT fleets	sensors, meters and cameras installed by the hundreds	no IP plan and no per-unit registration; different physical media in the same network
Edge computing and remote sites	plants, construction sites, areas with no stable coverage	the network works offline; allocation is local to the segment and several segments federate under one identity space
Cybersecurity and critical environments	networks where a rogue device is a physical as well as a digital risk	identities verifiable by construction, clone detection, native segmentation per customer or site, emergency isolation of the whole network
Time series and digital twins	long-term monitoring of assets whose hardware changes	the identity tells the asset from its physical incarnation: predictive maintenance and traceability with no external maps
Proximity and indoor	BLE localisation and tracking, personal devices	the same library runs on smartphones and microcontrollers and speaks Bluetooth and IP in the same network

The common thread is the same in every case: a network that builds itself, recognises its members and keeps working when the infrastructure is absent or unresponsive.

What has been measured

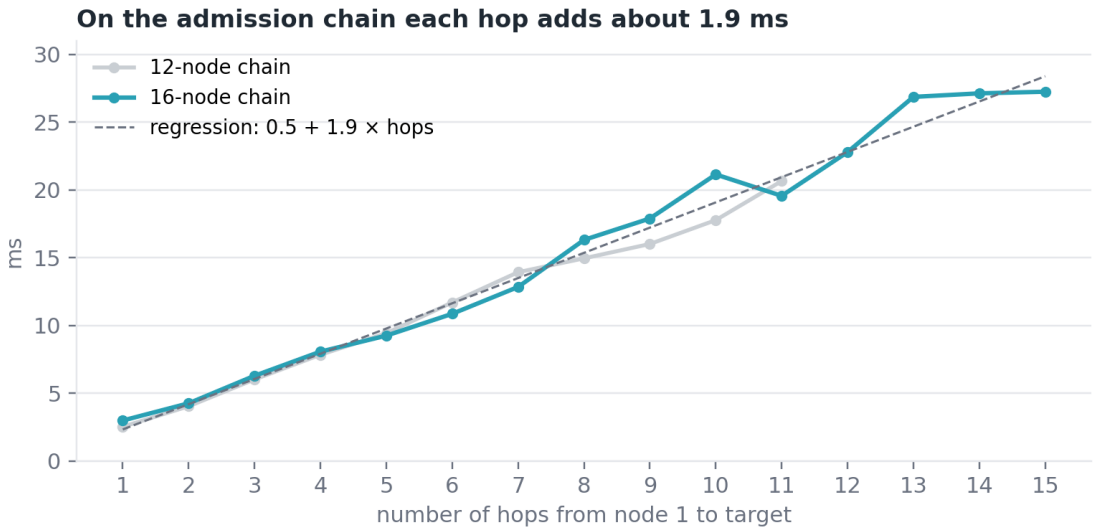
With libdaas 0.24.4 (test report of 1 November 2026) a full mesh is stable at 12, 16 and 20 nodes: no crashes, 100% probe delivery in all six runs, both when nodes probe one at a time (round-robin) and when all probe all at once (storm). Before the fixes of this version the same 12-node run crashed and delivered 27–42%.



Mean one-hop RTT, average of per-node values; 30 probes per peer, single host. Source: comparative RTT report 12/16/20 nodes, 1 November 2026.

The one-hop response time grows with the number of nodes because all processes run on the same host: it is a cost of the test environment more than of the protocol. Under concurrent load the time rises 2.8–2.9 times, but with no losses.

On a real admission chain (each node joins through the previous one), geometric routing delivers at every distance up to 15 hops and the time grows linearly, about 1.9 ms per hop: 3 ms at 1 hop, 27 ms at 15 hops. The only probe lost out of 1500 is the first one toward the direct neighbour.



Mean RTT by hop distance, 100 probes per target, join+CX chains of 12 and 16 nodes. Source: same report.

Current limits and next steps

The known limits are of capacity and maturity, not of model.

Limit	Effect	Next step
the IPv4 driver handles at most 20 connections per node	a direct full mesh reaches 21 nodes; beyond that, extra peers must be reached by routing	raise the limit or route toward peers beyond the twentieth
the DIN rank is 4 bits wide	an admission chain reaches 16 nodes (15 hops)	tree topologies with depth up to 15; format extension for longer lines
network admission is not authenticated	whoever takes part in the negotiation obtains a valid identity	bind the identity to a key (as IPv6 CGA do) and authenticate the join
duplicate detection not yet implemented	a conflict between rejoined segments is transient but not actively detected	verification at the allocator, as in 6LoWPAN
geometric routing only in desktop builds	embedded nodes interoperate but do not assign positions	bring the CX capability to ESP32 and Raspberry Pi
measurements on a single host	one-hop time grows with the number of processes on the same host	repeat the runs on separate hosts to separate protocol cost from environment cost

The version analysed is 0.24.4 with protocol 6.1.0; the developers guide and the test report are the reference documents of this overview.

Work in progress

Two evolutions of the model are under study and are only announced here; the technical details will be the subject of dedicated documents.

Security through vouching nodes. Network admission will be protected by a distributed trust model: a new node is admitted because other nodes already members, the vouching nodes, guarantee its identity. Trust resides neither in a server nor in a central authority, but propagates along the admission tree, by the same principle that today governs the assignment of identities. The goal is to close the limit noted above, unauthenticated admission, without introducing infrastructure.

Filtering of shared time (dATS). Time synchronisation between nodes will be refined with a filter that estimates and corrects over time the drift of each local clock with respect to the network reference. The expected result is a more stable network time and a lower maximum error, even over slow or irregular media such as LoRa and Bluetooth, without increasing synchronisation traffic.

Conclusions

daas lets the network build itself: identities, time and paths are not configured, they form with the network. It does not replace IP, cloud brokers or low-power networks; it covers what they do not: devices that must talk to each other even without infrastructure, over different physical media, with an installation cost per unit that tends to zero.

The measurements confirm that the model works in practice: a full mesh stable from 12 to 20 nodes with no losses, and multi-hop routing that delivers at every distance up to the protocol's depth limit with a linear cost of about 1.9 ms per hop. The known limits are of capacity and maturity, not of model, and the next steps, security through vouching nodes and the shared-time filter, address them while staying true to the same principle: no infrastructure, no central authority.

The scenario daas looks to is a pervasive network: thousands of devices in a city, in an industrial district or along a supply chain, recognising and talking to each other without anyone having registered them, even when the Internet link is missing or drops. Environmental sensors, meters, cameras, vehicles, robots and personal devices form a single network that grows as they switch on, reorganises when one switches off and keeps working by segments when isolated. In this scenario data no longer has to travel to a centre to be useful: it stays close to those who produce it and those who use it.

This is where daas meets the new paradigms of distributed artificial intelligence. Models running at the edge of the network, agents cooperating between neighbouring devices, federated learning between nodes that do not share raw data: all of this requires exactly what daas provides at its base, that is a way to know who a node is, when it was born, where it stands and what time it is, without a server certifying it. An identity verifiable by construction lets an agent trust what it receives; a shared time lets observations and decisions taken by different nodes be ordered; routing computed from the identity lets the right node be reached without a central map. In perspective, daas can be the communication fabric on which a distributed intelligence forms, coordinates and adapts, by the same principle by which the network forms: without infrastructure and without a central authority.

Contacts

Sebyone Srl · Via Pola, Cosenza · Piazza Capranica, Rome · Italy
sebyone.it · daasiot.com · info@sebyone.it · developers@sebyone.it
Paolo Lo Giacco (p.logiacco@sebyone.it) · Sebastiano Meduri (s.meduri@sebyone.it) · Lorenzo Grillo (l.grillo@sebyone.it) · Matteo Pagano (m.pagano@sebyone.it)

daas-api (libdaas) is released under the MPL-2.0 licence · © 2019–2026 Sebyone Srl. Document written on 3 November 2026 on libdaas 0.24.4, daas-protocol 6.1.0.