2019 International Conference on Networked Systems (NetSys)

VSL: A Data-Centric Internet of Things Overlay

Marc-Oliver Pahl, Stefan Liebald, Christian Lübben
Technical University of Munich
Email: {pahl,liebald,luebben}@s2o.net.in.tum.de

***Abstract*—Data-centric service-oriented designs are promising for overcoming the current IoT silos. The Virtual State Layer (VSL) is a data-centric middleware that securely unifies the access to distributed heterogeneous IoT components. The VSL solves key challenge of today's IoT: reducing the complexity, enabling interoperability, and providing security-by-design. The described practical setting enables the interactive exploration of a data-centric middleware including a live performance evaluation.**

***Index Terms*—data-centric, service-centric, Internet of Things, name-based, information-centric**

I. Introduction

The Internet of Things (IoT) provides an interface between our physical surrounding and the virtual world. By connecting distributed IoT devices, Pervasive Computing scenarios can be implemented. An example is a software application that dims the light up, opens the shutters, and makes breakfast at our desired wake up time [1].

Implementing similar scenarios is possible today. However, due to the distribution and heterogeneity of Things, it is complex. To manage the complexity, more recently service-oriented approaches were proposed [2], [3]. They modularize complex IoT applications into mashups of several more-simple and reusable microservices [4].

The IoT softwarizes our environments. The algorithms that orchestrate the Things operate on input and output data. With the Virtual State Layer (VSL), we present a data-centric service-oriented middleware for IoT orchestration.

Core tasks of a service-oriented data-centric middleware are *discovering*, *reading*, and *writing* data that belongs to other IoT service. In our example, the wake-up control service could write dim values into the data space of the lamp-control service in order to dim the light up.

The VSL associates structured semantic data with each IoT services that communicates over it. For different reasons including the providing of security-by-design [5], the VSL manages the data *for* services. In practice, a service registers to one of the VSL entry points that are called Knowledge Agent (KA). There it stores its data, and it uses the KA's programming interface (API) to couple with other services.

The described tuple-space communication [6] describes a paradigm shift from addressing Thing to addressing data. This is very similar to Information-Centric Networking (ICN), a clean-slate approach for Internet communication [7], [8]. Also, the VSL targets the management plane and not the data and control plane like fundamental ICN approaches [9]–[11].

Support: German Federal Ministry of Economic Affairs and Energy (BMWI) in DECENT (0350024A); German-French Academy SCHEIF.

Like [12], [13], the VSL implements ICN principles for enabling comprehensive orchestration of the IoT. However, the VSL does not propose replacing the Internet Protocol. Instead, it is implemented as site-local, self-organizing Peer-to-Peer overlay. The overlay enables co-existence-with and retrofitting-to existing infrastructures.

Different to the related works, the VSL manages the entire inter-service communication. This enables implementing desired properties such as high scalability, high performance, and a high level of security by-design [14] - meaning that the provided mechanisms cannot be circumvented. As such it is the first middleware that targets enabling a fully distributed service App development for the IoT [15], [16].

The VSL represents IoT devices and software as hierarchically structured data item graph. The data items can be *accessed transparently* from every participating VSL KA node. It offers different desired properties [17] including:

- a structured approach for representing IoT data [18]
- unified access to distributed service data (access transparency, location transparency) [19]
- late coupling of services via semantic discovery [20]
- security-by-design [5]

Section II introduces the system architecture focusing on overlay formation, data distribution, hierarchical addressing, structured data, data retrieval, and caching. Section III assesses the prototype's latency. Section IV demonstrates the use.

II. The data-centric VSL peer-to-peer overlay

Like other information-centric designs [12], [13], the VSL offers data access via `get`/ `set`, and publish/ subscribe. In addition, the VSL offers a synchronous coupling using streams over so called Virtual data Nodes [19].

As described before, the Application Programming Interface (API) of the VSL KAs is fixed. The variety between the managed IoT entities (hardware and software) reflects in their digital data twins. Our information model structures the digital twins as hierarchically structured data nodes.

VSL data nodes are tagged with data types (e.g. `integer`) and with functional identifiers (e.g. `lightDimValue`). To facilitate the modeling, we offer an object oriented approach that supports multi-inheritance [18]. To reach comparable expressiveness to ontologies [21] without introducing their complexity, we offer a modularized tagging approach [20].

When a service registers at a KA, it passes an identifier for a VSL data model. This model gets then instantiated in this KA and can be accessed by other services from that time on. Each service is represented as a data node tree.

Figure 1 shows a running VSL system. In the middle in the green VSL layer are the data model instances that the

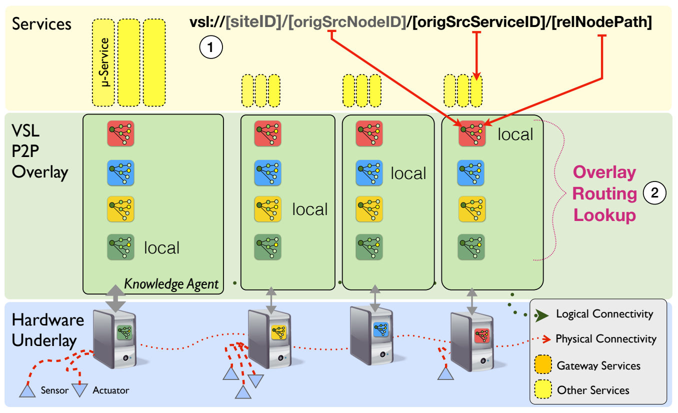

Fig. 1. The data-centric VSL peer-to-peer overlay with its addressing scheme (1) and current routing decision mechanism (2).

distributed VSL knowledge agents manage. The services on top are associated with their specific set of VSL data items each. The VSL runs on all IoT nodes with enough resources.

IoT data is not centrally stored but distributed. The data distribution in the VSL follows a *source principle*: data is always stored at the source. This is done as in contrast to typical Internet traffic, the IoT has dynamically changing inter-service communication relationships. This makes predicting cache locations difficult [22]. Since the VSL fosters service developers to store data in the KA instead of their service, and since data access to other services always happens through the connected KA, the source principle makes sense.

To enable unique addressing of the VSL peers, each node has a *locally unique identifier*. As each site also has a *globally unique identifier* [5], concatenating both leads to a globally unique identifier per VSL KA peer. When exchanging data between distributed IoT sites, this address can be used for global routing. Figure 1 shows the mapping of the different address parts to system features (1).

The VSL implements data discovery and routing as peer-to-peer overlay. The KA peers map between overlay IDs and the underlying substrate addresses that can be IP [17]. The discovery of data node identifiers happens via the tags [17], [20], e.g. `get/search/of-type/lamp` returns all data node instance addresses that are tagged with the type `lamp`.

The tag-based data node lookup happens KA-locally. The KAs regularly send pings over IP v6-multicast/ v4-broadcast to discover their peers. In addition, all site-local KAs synchronize, which data nodes they contain. This information includes the type and function types. Figure 1 shows this in the middle area. The different colors represent the different data sources.

Having all data locally enables a fast and resilient search for where data nodes can be retrieved. Together with the P2P node ID/ IP mapping, the corresponding KAs can be addressed. This mechanism works transparent for cached copies. However, knowing which data item is the most recent is an open challenge we are currently working on [17], [22].

Security is a key challenge of the IoT. IoT data is inherently threatening user privacy [23]. The VSL therefore protects all its components by adding a digital identity through X.509 certificates to it [24]. It also enables adding secured meta data [5]. Each VSL data node has identifiers set for read and write access. Those are matched with the identifiers that a service carries resulting in effective access control by-design [14].

III. Evaluation

Compared to direct service coupling, using the VSL for inter-service communication introduces latency via the KA processing. We therefore evaluate the added latency for accessing data of another service. For a more detailed usability, performance, scalability, and security see [17].

The access to VSL data nodes happens via `get`/ `set` on regular VSL nodes that are managed by the VSL KAs (regular coupling), and via direct function calls over virtual VSL nodes (virtual coupling) [19].

Latency is particularly relevant when mashing-up multiple IoT services, as it potentially adds up. Via the P2P overlay, all KA nodes are only one hop away.

In our measurement we assume full connectivity, making one overlay hop equal to one layer 2 hop. This does not necessarily reflect actual IoT deployments. However, as there is no IoT reference architecture, at least it enables comparable and reproducible measurements.

The same applies for our testbed resources. We used Intel Core i5 computers that are connected over 1GBps network links. This ensures that we do not run into a bottleneck with our measurement. The load on the nodes was always low.

For each test we measured 20000 independent *get* and *set* accesses on VSL data nodes. Table I shows the average latencies for the different coupling modes.

Operation	local		remote	
	regular	virtual	regular	virtual
get	1.3 ms	1.6 ms	10.4 ms	10.8 ms
set	1.9 ms	2.6 ms	9.3 ms	10.0 ms

Table I
Average delay of 20000 independent get/set requests.

The evaluation shows that the performance for requests on target services running on the same node is around 1.3-2.6 ms. For remote requests we achieve delays around 10 ms. The publish/ subscribe notifications on node data changes happen instantaneous, resulting in identical latencies to local accesses.

For the local and remote measurement under both coupling modes are low enough for mashing up to 20 always-remotely coupled, and about 70 locally running services. Such a complex application can still provide a real-time user experience.

IV. Demo

We demonstrate the data-based coupling and the service-orientation of the VSL. Our demo consists of two components:

- A smartphone based low-level controller.
- A light sensor based game.

Figure 2 gives an overview on our setting. The smartphone allows users to discover data items of the VSL type `light`. In our setting we have 2 alarm lights and 2 lights at our special game controllers. Via pre-formulated VSL queries, all found lights can be switched either by address or by type. This shows

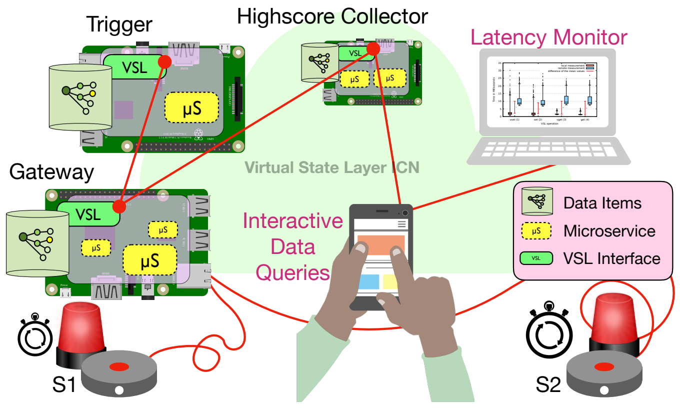

Fig. 2. The planned demo setup.

how the type-based discovery decouples services. It also shows the latencies of the operations.

The game part consists of multiple services. A *Gateway* interfaces our game controllers, a *Trigger* triggers randomly between 5-15s, and a *Highscore Collector* collects and displays the highscore. The game controllers consist each of an ambient light sensor that detects if covered or not, and LED indicators. The player's hands can cover the sensors.

When both controller's light sensors are covered, the game starts. Once the trigger fires, both controllers get the signal to switch their lights on. Then an internal timer starts that measures the time until the player's hand is removed from the light sensor. The measured time is provided to the *Highscore Collector*, and the winning controller blinks. Also the corresponding alarm light starts. If the player removes the hand before the light goes on, the controller reports ∞ as time and starts blinking immediately.

The Latency Monitor shows the most recent VSL queries and their latencies. As such, the perceived latency of the game and the quantitatively measured latencies give an overview on the VLSs latency.

Both setting run simultaneously, enabling interesting interaction as the smartphone can fire the game trigger.

V. Conclusion

The Virtual State Layer (VSL) middleware shows how the Internet of Things (IoT) can benefit from data-centric service-oriented orchestration. The implementation as peer-to-peer overlay allows running on top of the existing Internet while introducing the benefits of a new ICN principle-based design.

We introduced key mechanisms of the VSL, the modular modeling of digital twins, the globally unique addressing scheme, and the routing mechanism. We discussed our plans for using caching, and summarized the security-by-design properties. We evaluated the performance showing the effects of the implemented distributed data management. Our demonstration setting interactively illustrates the practical use and the properties of the system.

Our work shows how ICN principles can well be retrofitted into existing IoT networks. We hope that our work can contribute towards pushing the adoption of an integrated and secure IoT further into the real world.

References

[1] M. Weiser, "The Computer for the 21st Century," *Scientific American*, vol. 265, no. 3, pp. 94–104, Sep. 1991.

[2] D. Lu, D. Huang, A. Walenstein, and D. Medhi, "A Secure Microservice Framework for IoT," in *2017 IEEE Symposium on Service-Oriented System Engineering (SOSE*. IEEE, 2017, pp. 9–18.

[3] M.-O. Pahl, G. Carle, and G. Klinker, "Distributed Smart Space Orchestration," in *Network Operations and Management Symposium 2016 (NOMS 2016) - Dissertation Digest*, 2016.

[4] M.-O. Pahl, H. Niedermayer, H. Kinkelin, and G. Carle, "Enabling Sustainable Smart Neighborhoods," in *3rd IFIP Conf. on Sustainable Internet and ICT for Sustainability (SustainIT)*, Palermo, Italy, 2013.

[5] M.-O. Pahl and L. Donini, "Giving iot edge services an identity and changeable attributes," in *International Symposium on Integrated Network Management (IM)*, Washington DC, USA, Apr. 2019.

[6] D. Gelernter, "Generative communication in Linda," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 1985.

[7] A. Lindgren, F. B. Abdesslem, B. Ahlgren, O. Schelén, and A. M. Malik, "Design choices for the IoT in Information-Centric Networks," *13th Consumer Communications and Networking Conference (CCNC)*, pp. 882–888, 2016.

[8] M. Amadeo, C. Campolo, A. Iera, and A. Molinaro, "Named data networking for IoT: An architectural perspective," *EuCNC 2014 - European Conf. on Networks and Communications*, no. July 2015, 2014.

[9] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, "A data-oriented (and beyond) network architecture," *ACM SIGCOMM Computer Communication Review*, 2007.

[10] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *5th int. conf. on Emerging networking experiments and tech.* ACM, 2009.

[11] B. Ahlgren, V. Vercellone, M. D'Ambrosio, M. Marchisio, I. Marsh, C. Dannewitz, B. Ohlman, K. Pentikousis, O. Strandberg, and R. Rembarz, "Design considerations for a network of information," *Proceedings of the 2008 ACM CoNEXT Conference on - CONEXT '08*, 2008.

[12] S. Chatterjee, "A Survey of Internet of Things (IoT) over Information Centric Network (ICN)," no. August, pp. 0–18, 2018.

[13] S. Arshad, M. A. Azam, M. H. Rehmani, and J. Loo, "Recent Advances in Information-Centric Networking based Internet of Things," *IEEE COMM. SURVEYS & TUTORIALS*, 2018.

[14] A. Cavoukian, "*Privacy by Design*: Leadership, Methods, and Results." *European Data Protection*, pp. 175–202, 2013.

[15] M.-O. Pahl, "Multi-tenant iot service management towards an iot app economy," in *HotNSM workshop at the Int. Symposium on Int. Network Management (IM)*, Washington DC, 2019.

[16] M.-O. Pahl and G. Carle, "Taking Smart Space Users into the Development Loop: An Architecture for Community Based Software Development for Smart Spaces," in *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*. New York, NY, USA: ACM, 2013, pp. 793–800.

[17] M.-O. Pahl and S. Liebald, "Designing a Data-Centric internet of things," in *2019 International Conference on Networked Systems (NetSys) (NetSys'19)*, Garching b. München, Germany, Mar. 2019.

[18] M.-O. Pahl and G. Carle, "Crowdsourced Context-Modeling as Key to Future Smart Spaces," in *Network Operations and Management Symposium 2014 (NOMS 2014)*, May 2014, pp. 1–8.

[19] M.-O. Pahl, "Data-Centric Service-Oriented Management of Things," in *Integrated Network Management (IM), 2015 IFIP/IEEE International Symposium on*, Ottawa, Canada, May 2015, pp. 484–490.

[20] M.-O. Pahl and S. Liebald, "A modular distributed iot service discovery," in *International Symposium on Integrated Network Management (IM)*, Washington DC, USA, Apr. 2019.

[21] U. Aßmann, S. Zschaler, and G. Wagner, "Ontologies, Meta-models, and the Model-Driven Paradigm," in *Ontologies for Software Engineering and Technology*, C. Calero, F. Ruiz, and M. Piattini, Eds. Berlin Heidelberg: Springer, 2006, pp. 249–273.

[22] M.-O. Pahl, S. Liebald, and L. Wüstrich, "Machine-learning based IoT Data Caching," in *Integrated Network Mgmt. (IM), 2019 HotNSM at IFIP/IEEE International Symposium*, Washington, USA, 2019.

[23] M.-O. Pahl and F.-X. Aubet, "All eyes on you: Distributed Multi-Dimensional IoT microservice anomaly detection," in *14th Int. Conf. on Network and Service Management (CNSM)*, Rome, Italy, Nov. 2018.

[24] M.-O. Pahl and L. Donini, "Securing IoT Microservices with Certificates," in *Netw. Operations and Man. Sym. (NOMS)*, Apr. 2018.