# Smart-M3 vs. VSL for IoT

Ilena Pesheva, Christian Lübben*
*Chair of Network Architectures and Services, Department of Informatics
Technical University of Munich, Germany
Email: pesheva@in.tum.de, luebben@net.in.tum.de

***Abstract*—The benefits of autonomous data exchange in software environments and multi-device communication have triggered the development of various middleware services to enhance data exchange in distributed systems. These middleware solutions have gained great importance in reducing overall system complexity and enabling interoperability in the context of information sharing.**

**This paper takes a closer look at two data exchange middleware solutions: The Smart-M3 platform and VSL overlay for Internet of Things. They both solve key challenges in the environment of device heterogeneity and propose a data-centric approach to information exchange.**

**We explain the core differences in their key features and give an overview of their field of application.**

***Index Terms*—middleware, data-centric, interoperability**

## 1. Introduction

In the era of rapid digitalization and constant emerging of new technological devices, the idea of seamless and wireless information exchange between these devices has evolved to a need. This is referred to as ubiquitous computing, which is gradually emerging as the dominant type of computer access [1]. As M. Weiser states in [1], it is enabling nothing fundamentally new, but by making everything faster and easier to do, it is transforming the perception of what is possible. An example is the Internet of Things (IoT) that quickly gained importance as part of the Internet mainly because of its immense impact, remarkable growth and capability. Its main features are interrelating "computing devices, mechanical and digital machines, objects, animals or people that are provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction" [2]. Benefiting from ICN-principles IoT has a data-centric traffic design rather than addressing a specific host. This results in improvements in data latency, scalability, reliability, resilience and is more energy efficient than typical host-based communication [3].

The distributed nature of the devices participating in an IoT system and the heterogeneity of application scenarios introduce interoperability problems and challenges for developers. This triggered the introduction of the Virtual State Layer (VSL). It is, as M.-O. Pahl, S. Liebald, and C. Lübben specify in [3], "a data-centric middleware that securely unifies the access to distributed heterogeneous IoT components.". Its core tasks are to discover, read and write data that another IoT component has produced and thus to be able to orchestrate IoT environments.

The idea behind it is Service Oriented Architecture (SOA). It modularizes the complex IoT applications into smaller mashups, creating microservices that are simple and can be later reused. It fully separates logic and data in IoT services which contributes to major VSL features such as delivering service data even when a service is offline or security-by-design, meaning that security is fully implemented within the VSL [4].

The interest of getting computing devices to interoperate and to do so with whatever devices are locally close at any point in time has also raised the question of dealing with complexity and interoperability issues. Participating in different domains means implementing several different standards, which are often serving specific use cases and not aiming to create a general interoperability framework [5]. The semantic web is attempting to defy this issue by implementing an interoperability framework with the ultimate goal of making a machine understand the World Wide Web data. The result would be a "giant global graph" of linked data that describes the information of the web in a standard model for data interchange - Resource Description Framework (RDF) [6] and Web Ontology Language (OWL) [7] that enable the encoding of semantics [8]. Nonetheless, there is a need for a mechanism that enables sharing of dynamic, rapidly changing information on a local level about the current state of a device and the web is not the most suitable environment for that purpose. Therefore, the interoperability platform Smart-M3 was created with the goal, as explained in [5], "to (enable devices) to easily share and access local semantic information, while also allowing access to the locally relevant parts of the 'giant global graph' to be available". Smart-M3 acts as a smart space solution that enables devices of all kinds to engage in interoperability and have a shared view of data and services. The goal is to provide a better user experience where users can effortlessly add or remove devices from the platform and grant every participating device the same information access.

The similarities between the VSL and Smart-M3 platforms in their main purpose and idea raises the importance to differentiate them and thus to be able to apply them accordingly.

## 2. Feature comparison

Both VSL and Smart-M3 platforms serve as distributed systems middleware and share the common purpose of enabling interoperability in heterogeneous environments. However, they rely on different architectural approaches and comprehend data in a different manner. The following section gives a brief overview of the general system architecture of Smart-M3 and VSL. Furthermore, we address their core differences considering the following key feature components: data access, data storage, data discovery, data transport, semantic structure and security. This step-by-step comparison will allow us to obtain a full perspective of the functional and semantic divergence of each system.

## 2.1. General system architecture

Smart-M3 works on the basis of a blackboard architectural model and implements the idea of space-based computing. The architecture it implements consists of two core components: knowledge processors (KP) and semantic information brokers (SIB) that may be concrete or virtual. The core of the system is hosted by a device which contains the SIB and the physical data base. Then there are other devices hosting KPs - pieces of software implemented to read and contribute data to a SIB [9]. One or more SIBs connected to each other define a smart space that contains information provided by the KPs. An illustration example of a M3 smart space distribution is presented in Figure 1 featured in [10] by J. Honkola, H. Laine, R. Brown and O. Tyrkkö.

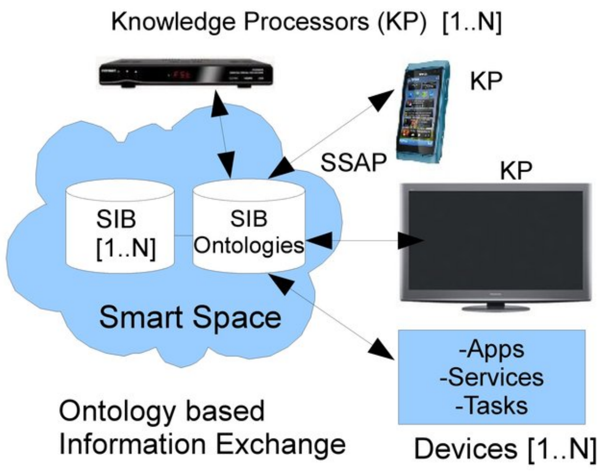

Figure 1: Smart-M3: SIB and KP distribution

The VSL overlays key components are the so-called Knowledge Agents (KA) that manage data for different services. Every KA serves the purpose of storing relevant data for a specific node while also enabling inter-service communication [3]. Figure 2 described in [3] gives a detailed view of the VSL architecture model, consisting of a hardware underlay with IoT nodes, VSL Peer-to-Peer overlay and multiple microservices that register at a KA. The VSL offers IoT nodes unified access to the Knowledge Agents, which can also run on the same IoT node like a service [4].

## 2.2. Data access

The connection between the SIBs in Smart-M3 is enabled by a protocol providing distributed deductive closure

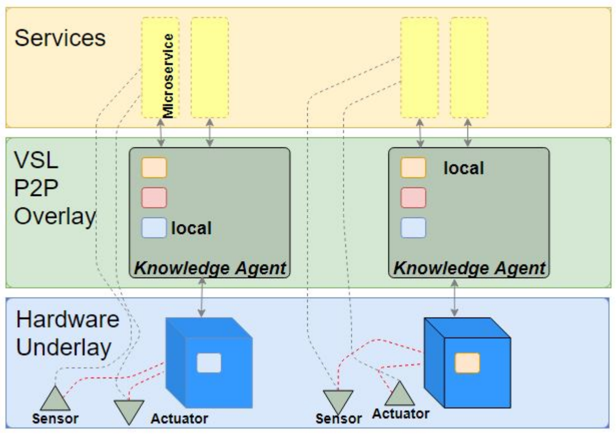

Figure 2: VSL general architecture and logical connectivity

[11]. This allows all KPs to access the same information in the smart space with no regard of the specific SIB they are connected to.

[5] A KP can access a SIB by using Smart Space Access Protocol (SSAP). This protocol has eight operations: *join, leave, insert, remove, update, query, subscribe* and *unsubscribe*. Therefore, the KPs are able to interact with the content in the smart space. The operations enabling this interaction are not concrete as they depend on the defined parameters and the actions that SIB and KP should initiate. They may also be encoded using different formats, JSON or XML for example.

[5] The protocol uses sessions to establish a connection between the KP and the SIB components. First, a *join* operation is executed by the KP, the SIB then inspects it and decides whether the KP can join or not. Following a successful join the KP is allowed to perform other activities. If the SSAP protocol is successfully supported by the SIB and KP implementations interoperability will be ensured.

The VSL, as mentioned in the Introduction, finds its specific purpose in Internet of Things services. Its working principle is to fully decouple data from hosts providing ICN properties [12]. Unlike Smart-M3, it is implemented as self-organizing Peer-to-Peer overlay, enabling data-centric communication between and within IoT hosts. It targets microservice-based architecture in order to run independent IoT microservices. Access is based on hierarchical data item identifiers and is enabled via get and set, subscriptions to changes and stream connections [4]. This is to be differentiated from the Smart-M3 data access method, which is push-based one instead of a publish-subscribe one as in VSL.

## 2.3. Data Storage

Smart-M3, as already stated in Section 2.1, has two key components - the knowledge processor and the semantic information broker. The information is stored in the smart space, consisting of one or more SIBs, as an RDF graph. The storage is realized on some defined ontology, however, there is no obligation for using a specific one. The KPs, having once successfully accessed the smart space, are then able to contribute to or to read the stored in the smart space content. [5]

VSL works on a different principle for storing data (see Figure 2). Instead of having a distributed shared memory architecture that contains all the information like the smart space in the Smart-M3 platform (see Figure 1), it stores data always at the source. This follows as a consequence of the disconnection between logic and data [4]. The platform implements data managing agents, the KAs, which are connected to each other in a Peer-to-Peer structure. Each of them is responsible for running a number of microservices and stores data relevant to them. This results in distributing information to be stored and retrieved all over the network. Therefore, locality is an important issue when dealing with VSL, especially when information exchange happens constantly. However, due to the full location transparency in the data lookup process, data can be stored on any KA, not necessarily on the one that is running on the same IoT node (source) as the service.

## 2.4. Data Discovery

The VSL, as already mentioned in Section 2.1, implements data discovery in a Peer-to-Peer manner between the KA peers. The KAs are assigned with overlay IDs and an underlying address that can be IP-Address. The data node discovery happens via special tags, provided by the KAs, which return all instance addresses associated with the given tag. This means that the semantic lookup happens KA-locally via a search for coupling candidates. Services do not bind statically, unlike how the Knowledge Processors in Smart-M3 bind to their services.

In contrast to the encapsulated nature of the VSL data discovery that is fully integrated and closed to the concrete IoT network, the Smart-M3 platform is allowed access to parts of the "giant global graph" that results from the Semantic Web [8], in addition to sharing and accessing local semantic information between the engaged software entities and devices. Thus, it is making use of both local and global information, represented as an RDF graph. This allows easy linking of data between different ontologies, which aims to solve the interoperability issue. Unlike the Semantic Web, which represents the idea of a single, centralized web of machine-understandable information, Smart-M3 sets distinct spots, the Knowledge Processors, in the Web. These spots may be connected to many devices of different kinds and gather specific machine-understandable information that is unique but non-exclusive for the particular KP and has a concrete focus and purpose. Overlapping of information between the KPs is even needed to ensure interoperability [5].

## 2.5. Data Transport

[5] In the Smart-M3 interoperability platform, the core component responsible for data interaction is the SIB. Its internal architecture consists of five layers: Transport layer, Operation handling layer, Graph operations layer, Triple operations layer and Persistent storage layer. The transport layer, being the first access point of the SIB architecture, must ensure that various domains, service architectures and networks are able to communicate and exchange information, regardless of their different communication capabilities. To be able to overcome this issue, the SIB supports various communication mechanisms, such as Bluetooth, TCP/IP, HTTP and NoTA. The most suitable mechanism is being selected depending on the operating environment. This indicates once more one of the core principles in Smart-M3 - to be able to operate regardless of the communication mechanisms restrictions.

[4] As already mentioned in Section 2.3, VSL organizes its information exchange in a Peer-to-Peer process by distributing it between a number of Knowledge Agents. Thus they must be capable of addressing each other accordingly. The information exchange is enabled via IP unicast and multicast connections. Unicast is used in the case of a single recipient of the data and multicast - as the core maintenance of the overlay. The Knowledge Agent has a Transport Manager, which along with the Connection Manager and the Overlay Manager, manages the connectivity between the IoT nodes. It uses HTTP over TCP/IP as a transport protocol, although different protocols are also applicable.

In contrast to Smart-M3, VSL does not adopt any other data transport technology, such as Bluetooth for example, as the communication transport happens within the VSL overlay. It does not have the need to accommodate to the different capabilities of other domains or service architectures like Smart-M3 does.

## 2.6. Semantic Structure

[5] The Smart-M3 platform allows storing and querying information on the basis of tuple space mechanisms. This means that data is exchanged between a consumer and a producer entity. In the case of Smart-M3 these are the SIB and the KP respectively. Data is produced in tuple form and retrieved from the consumer using a specific pattern. As mentioned in Section 2.5, there are five logical layers responsible for the access, operations and storage of data to the SIB. After access has been established, requests in the form of SSAP operations run in threads to query, insert or remove information from the RDF store [6]. This is handled by the Graph operations layer where the operations are being scheduled. In the Triple operations layer happens the inserting, querying and removing of triplets from the RDF store. Triplets, connected to form a graph, represent the architecture or the RDF data format. The linking of triplets results in a structured data graph, resembling the World Wide Web. A triple is a statement in the form subject-predicate-object (e.g the sky /subject/ has the color /predicate/ blue /object/). The RDF syntax is abstract, meaning that it can be represented by using a variety of notations in the arrangement order subject-predicate-object. The semantic is the representation of the subject, predicate and object roles in the statement. Due to the RDF format of data its linking under different ontologies has been made extremely easy, reducing system complexity and enabling cross-domain interoperability.

[3] Similarly to the Smart-M3 platform, VSL also implements a tuple-space mechanism in the form of a structured data item graph to organize its data, although in a hierarchical manner. Each time a service is being registered at a Knowledge Agent, an identifier is passed for the data model representation. An instance of this

model is being created at the KA allowing the connection and communication of this service to other services via the KA API. The data nodes participating in the structured graph are in fact the digital data twins of the managed IoT software and hardware entities. VSL uses tags and identifiers pinned to the data nodes, offering a modularized tagging approach. The items in the hierarchical structure can be accessed transparently from an arbitrary participating KA.

To summarize, Smart-M3 is using data in the form of triplets according to the RDF syntax to store and retrieve information, whereas the VSL uses a hierarchical data structure in the form of a data node tree and is instantiating digital twins of data every time a service and a KA interact.

## 2.7. Security

[4] The main issue of the IoT data is security because of the vulnerable nature of private user data. Therefore, VSL implements security-by-design, which means that the mechanisms implemented in the VSL middleware cannot be outmaneuvered. This, as mentioned in Section 2.1, comes as a result of the full separation of service logic and data and promises a secure throughout communication. In particular, VSL assigns certificates to each of its components (services and KAs). These certificates ensure the authentication of software modules, as well as enabling communication between KAs that is TLS-secured, including secure exchange of keys for encrypted stores. Due to access control to IoT nodes and specific synchronization of type information and access modifiers between the KAs, VSL ensures secure addressing and trusted IoT orchestration.

The smart space environment is vulnerable to threats and security risks as well. In contrast to VSL, Smart-M3 has not been provided a sufficient security mechanism to this point. In [13], Kirill Yudenok and Ilya Nikolaevskiy introduce a security solution protecting the data interchange between the KPs and the smart space. For robust authentication they propose the usage of the Host Identity Protocol (HIP) for key exchange [14]. The HIP protocol can be integrated in the SIB access module and thus enable a SIB to restrict access to information in the smart space that the KPs have provided.

## 2.8. Comparison summary

In order to retain a clear and structured view of the information presented in the previous subsections, Table 1 summarizes and highlights the most important aspects and differences of the Smart-M3 and VSL middlewares.

## 3. Issues and Reliability

Both the Smart-M3 platform and the VSL overlay aim to implement simplistic architecture designs for reliability and robustness reasons. However, issues and challenges are an inevitable part of every system regardless of its kind. In the following we highlight some important aspects to consider when dealing with data-centric issues and vulnerabilities of each system.

TABLE 1: Key feature differences

| | **Smart-M3** | **VSL** |
|---|---|---|
| **Data access** | SSAP | get/set |
| **Data storage** | smart space | at the source |
| **Data discovery** | use of local or global information | tag or address based |
| **Data transport** | Bluetooth, TCP/IP HTTP and NoTA | HTTP over TCP/IP |
| **Data structure** | RDF graph | hierarchical graph |
| **Security** | not implemented internally | security-by-design |
| **Implementation** | C, C++, Python, C#, Java | Java |

## 3.1. Smart-M3

Arguably the main issue in the Smart-M3 system is security, as mentioned in Section 2.7. There we elaborated the main causes of this issue and mentioned a resolution method presented by K. Yudenok and I. Nikolaevskiy in their work [13]. In [15] written by Matti Eteläperä et al. a test of two smart space information broker implementations is presented: Smart-M3 and RIBS (RDF Information Based System), the second being an M3 tool for devices with restricted computational capabilities. Based on their measurement analysis the authors conclude that neither system is satisfactory enough for wide-spread usage. The authors state that "Smart-M3 performance and usability leave a lot to be desired, as even a simple single triple insert operation has a latency of 86-176 milliseconds in our tests. The performance of Smart-M3 is not suitable for use cases needing fast response times."

## 3.2. VSL

The Virtual State Layer decouples data not only from hosts (ICN principle), but even from services on hosts, as already mentioned in the Introduction. This presents a significant advantage in the case of a service failure, as decoupled data is then managed only within the middleware. Energy efficiency also follows as a consequence, because not needed services can be interrupted while their data still remains attainable. Nevertheless, the advantages that follow from the ICN principle have challenges on their own. In [16] the authors A. Lindgren, F. B. Abdesslem, et al. address the aspects of naming, caching, actuation, decoupling between publisher and consumer, etc. They suggest that naming can become a size-problem when e.g. the size of the name can become larger that the size of the data; Caching reduces latency but can also be useless when using At Most Once object requesting strategy; Actuation may conflict with the ICN addressing design and further reduce caching advantages and impose latency requirements; Although having multiple advantages, decoupling publisher/consumer has trouble in resolving publisher mobility when deducing the name of the data for consumers.

## 4. Field of application

In [5] the authors present several smart space application scenarios. They involve different applications, various services and multiple types of devices (e.g. phones, laptops, sensors). All domains have M3 software agents installed on them. An example described in [17] shows a scenario that involves an application for sports tracking, a music streaming service, a gaming domain and a phone call observer application. The results show improved user experience due to seamless component cooperation between the participating devices and services. An ongoing call can trigger information exchange in the smart space resulting in pausing the music and the game. Furthermore, one example demonstrates a mash-up between two different scenarios, proving the ease of their mixing. [5]

An example for the VSL overlay in use is shown in [3] by Marc-Oliver Pahl, Stefan Liebald and Christian Lübben. In their work they present a VSL demo consisting of a smartphone based controller and a light sensor based game whereby they "demonstrate the data-based coupling and the service-orientation of the VSL" [3]. Each participating service implements a VSL interface, several microservices and local data items (figure 2). Users make interactive data queries via the smartphone controller and thus switch lights found by type or address.
This demo illustrates the benefits of VSL in environments where decoupling specific services is needed, whereas the Smart-M3 platform, while also demonstrating inter-service communication, points out the ease of creating scenario mashups.

## 5. Conclusion

This paper reviews the similarities and differences between the Smart-M3 platform and the VSL IoT overlay. Both systems aim to solve the interoperability problem and thus present data-centric solutions for reducing system complexity in heterogeneous environments. However, they employ contrasting approaches and architectural styles, which makes their distinction of great importance. Smart-M3 is a space-based system that enables interoperability through the use of the Semantic Web. It targets multi-device implementation and enables sharing of local information between hosts. VSL follows ICN principles and has its main focus on Internet-of-Things environments where it manages the entire inter-service communication between IoT devices. It implements a Peer-to-Peer architecture for routing and unlike Smart-M3, VSL is making use of the network connectivity between devices to enable communication and data exchange. It implements security-by-design, whereas Smart-M3 has not implemented a specific security mechanism. Thus we have shown that Smart-M3 can be used easily in an environment of different domains; it handles usage and mixing of various scenario instances. The VSL overlay has its specific focus on the IoT environment where it manages each aspect of the communication services.

## References

[1] M. Weiser, “The computer for the 21st century,” vol. 3, no. 3, pp. 3–11. [Online]. Available: https://doi.org/10.1145/329124.329126

[2] What is internet of things (IoT)? - definition from WhatIs.com. (Date accessed: 30.11.2019). [Online]. Available: https://internetofthingsagenda.techtarget.com/definition/Internet-of-Things-IoT

[3] M.-O. Pahl, S. Liebald, and C. Lübben, “VSL: A data-centric internet of things overlay,” in *2019 International Conference on Networked Systems (NetSys)*, pp. 1–3.

[4] M.-O. Pahl and S. Liebald, “Information-centric IoT middleware overlay: VSL,” in *2019 International Conference on Networked Systems (NetSys)*. IEEE, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/8854515/

[5] J. Honkola, H. Laine, R. Brown, and O. Tyrkkö, “Smart-m3 information sharing platform,” in *The IEEE symposium on Computers and Communications*, pp. 1041–1046, ISSN: 1530-1346.

[6] RDF - semantic web standards. (Date accessed: 12.01.2019). [Online]. Available: https://www.w3.org/RDF/

[7] “OWL web ontology language overview,” p. 22, (Date accessed: 19.02.2020). [Online]. Available: https://www.w3.org/TR/owl-features/

[8] T. Berners-Lee, J. Hendler, and O. Lassila, “Scientific american: Feature article: The semantic web: May 2001,” p. 4, (Date accessed: 23.02.2020). [Online]. Available: https://www-sop.inria.fr/acacia/cours/essi2006/Scientific\%20American_\%20Feature\%20Article_\%20The\%20Semantic\%20Web_\%20May\%202001.pdf

[9] I. Oliver, “M3 information SmartSpaces technology overview,” (Date accessed: 05.12.2019). [Online]. Available: https://www.slideshare.net/ianoliver79/m3-information-smartspaces-technology-overview

[10] SOFIA - smart m3 information-sharing platform. NOKIA. (Date accessed: 29.01.2020). [Online]. Available: https://www.slideshare.net/sofiaproject/sofia-smart-m3-informationsharing-platform

[11] “A mechanism for managing and distributing information and queries in a smart space environment:,” in *Proceedings of the Joint Workshop on Advanced Technologies and Techniques for Enterprise Information Systems*. SciTePress - Science and and Technology Publications, pp. 145–153. [Online]. Available: http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0002193101450153

[12] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, “A survey of information-centric networking,” vol. 50, no. 7, pp. 26–36. [Online]. Available: http://ieeexplore.ieee.org/document/6231276/

[13] K. Yudenok and I. Nikolaevskiy, “Smart-m3 security: Authentification anc authorization mechanisms,” in *2013 13th Conference of Open Innovations Association (FRUCT)*, pp. 153–162, ISSN: 2343-0737.

[14] A. Gurtov, M. Komu, and R. Moskowitz, “Host identity protocol: Identifier/locator split for host mobility and multihoming,” (Date accessed: 15.02.2020). [Online]. Available: https://www.researchgate.net/publication/233893326_Host_identity_protocol_Identifierlocator_split_for_host_mobility_and_multihoming

[15] M. Etelapera, J. Kiljander, and K. Keinanen, “Feasibility evaluation of m3 smart space broker implementations,” in *2011 IEEE/IPSJ International Symposium on Applications and the Internet*, pp. 292–296.

[16] A. Lindgren, F. B. Abdesslem, B. Ahlgren, O. Schelén, and A. M. Malik, “Design choices for the IoT in information-centric networks,” in *2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC)*, pp. 882–888, ISSN: 2331-9860.

[17] J. Honkola, H. Laine, R. Brown, and I. Oliver, “Crossdomain interoperability: A case study,” in *In Smart spaces and*, pp. 22–31.