## MicroServices (MS)

Microservices are an architectural and organizational approach to software where

software produces or consumes small independent microservices. A produced microservice can be consumed by a microservice consumer.

Such service consumption constitutes an exchange of data using well-defined service interfaces/APIs. Such services are hosted by/owned by small self-contained microsystem. A microsystem normally host a small (<5) number of microservice.

Each service runs in its own process. The services communicate with clients, and often each other, using lightweight protocols, often over protocols like e.g. MQTT, CoAP, HTTP, Websocket, DDS

From a number of microsystems and their microservices a System of Systems (SoS) is composed by orchestrating service exchanges between the microsystems. This is the foundations for a service oriented architecture, SOA.

## Service oriented and microservice architecture

Service-oriented architecture, SOA, introduced by IBM in the mid 1990s[[1]](#footnote-1), is often referred to as micro-service architecture. From a more fundamental perspective, these terms are considered equivalent[[2]](#footnote-2). In this context, a micro-service is produced/consumed by a micro-system. A SOA reference model has been published by OASIS[[3]](#footnote-3). The key properties of SOA/micro-service architectures are:

– Look-up  
– Late binding  
– Loose coupling

often referred to as the 3 Ls.  
Further, important properties are as follows:

– A micro-system performs its function independently.   
– A micro-system can be   
 • stateful and is then responsible for storing its own state; or   
 • stateless.

Fulfillment of these properties requires a control plane composed of at least a service registry and an orchestration capability. Today this is most often complemented by an authorization and authentication capability.

A prominent early usage of these fundamentals was the Jini add-on package to Java[[4]](#footnote-4). SOA is currently the main approach for achieving automa- tion and digitalization architectures and their reference implementations and platforms for concrete solution imple- mentation. Examples include FiWare[[5]](#footnote-5), Eclipse BaSyx[[6]](#footnote-6), and Eclipse Arrowhead[[7]](#footnote-7)[[8]](#footnote-8). A comparison of the current most relevant initiatives related to automation and digitalization can be found in[[9]](#footnote-9).

For the implementation of SOA, there is a wide range of protocols that can be considered service protocols, e.g., HTTP (REST), CoAP, MQTT, DDS, and AMQP[[10]](#footnote-10) [16].

## Microservice architectures building blocks

SOA is one approach to integrate automation and digitalization solutions based on a set of heterogeneous Microsystems and their microservices. This enable for example solution integration following Industry 4.0 architectures such as RAMI4.0[[11]](#footnote-11) and IIRA[[12]](#footnote-12). Their implementations are often achieved using an SOA/micro-service approach and principles, properties and terminologies based on SoS and the IoT. Some SoS fundamentals have been formulated by Maier[[13]](#footnote-13) [[14]](#footnote-14) and then updated and extended by Boardman and Sauser[[15]](#footnote-15). A further extension to the biology of systems has been proposed by Sauser et.al.[[16]](#footnote-16). Therefore, the SoS properties that must be captured in modeling can be summarized as follows:

* *Operational independence/autonomy of the elements* The constituent systems can operate independently in a meaningful way and are useful in their own right.
* *Belonging* The autonomous constituent systems choose to belong to the SoS because they see value for themselves to give up some of their autonomy to receive benefits.
* *Connectivity* To let the constituent systems interact, they must be connected, and unless they provide sufficiently generic interfaces, they must be modified to provide such interoperability. Connectivity in an SoS is, thus, dynamic, with interfaces and links forming and vanishing as the need arises.
* *Diversity* Whereas many other systems strive to minimize diversity to simplify the system, increased diversity in an SoS gives it the ability to better address unforeseen situations during its life-cycle.
* *Emergent behavior* appears in any system, and in many systems, this is deliberately and intentionally designed in and tested. In an SoS, emergent behavior is not restricted to what can be foreseen. Instead, the system should have the capability to detect (in an early stage) and eliminate bad behavior that emerges.
* *Managerial independence of the elements* The con- stituent systems not only can but do operate independently, even while being part of the SoS. Further, elements can be added individually to the SoS.
* *Evolutionary development* The SoS does not appear fully formed, and functions and purposes are added based on experience.
* *Geographical distribution* The constituent systems only exchange information and not substantial quantities of mass or energy.
* *Secure and safe* Malicious behavior in an SoS and its constituent systems must be detected and mitigated to ensure information, system and SoS integrity.

The literature clearly points out that SoS as defined by Maier can be implemented using SOA/microservice architectures technology[[17]](#footnote-17).

To address SoS requirement like e.g. real time, safety, security, validation and verification etc. segmentation of microsystems and their service becomes necessary. This can be addressed by the concepts of local clouds and system of local clouds[[18]](#footnote-18). The local cloud ans system of local clouds concept can be seen a building blocks in complex SOA/microservice architectures.

## SOA/microservice architecture Building Blocks

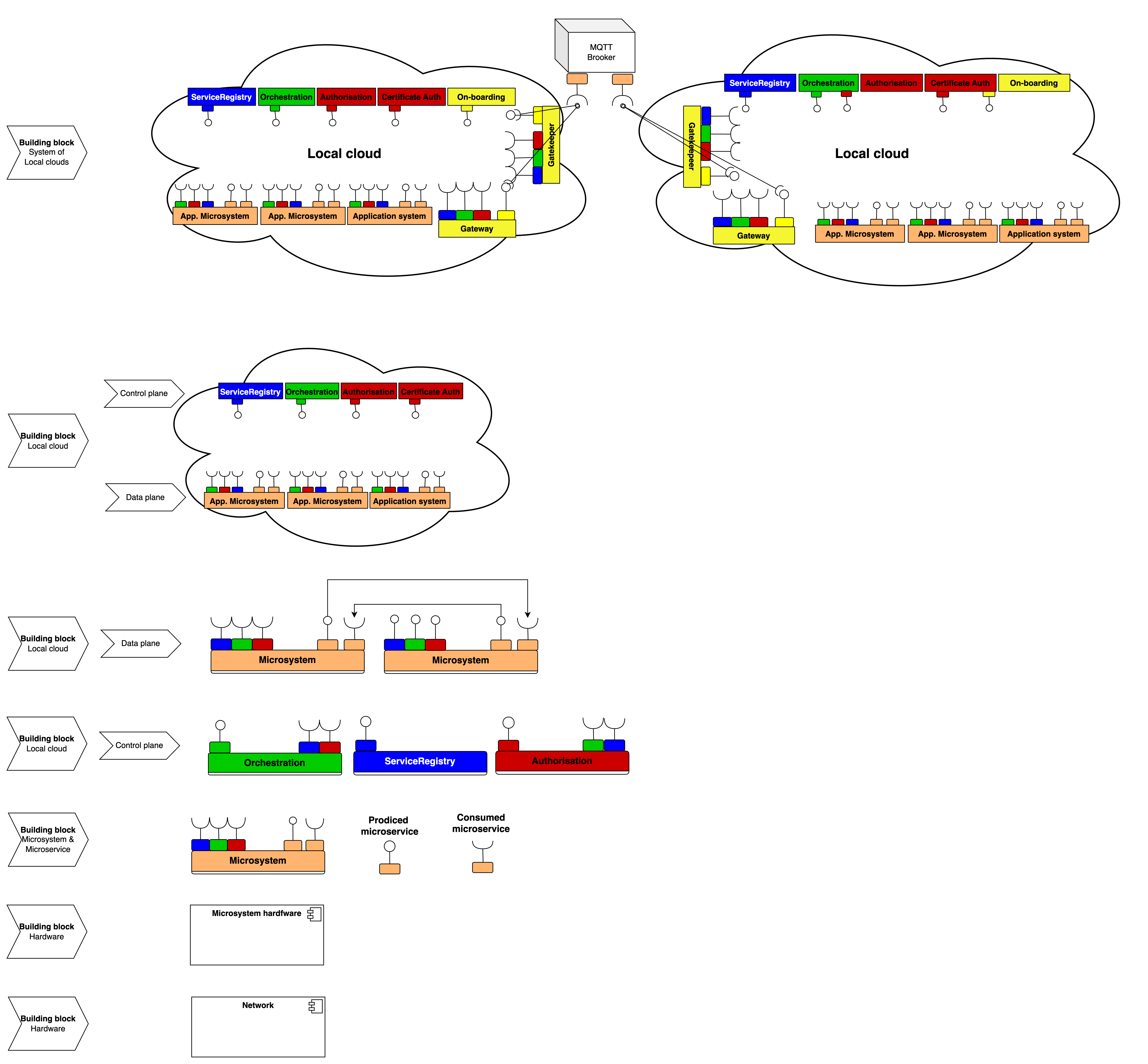
Based on above definitions we can define set of SOA/microservice architecture layers or building blocks se figure 1 below. In below figure a set of building blocks based on microservice, microsystem and SOA has been identified.

Figure 1: Basic building blocks in an microservice, microsystem and SOA paradigm.

1. Erl T (2005) Service-oriented architecture: concepts. Prentice Hall, Technology and Design [↑](#footnote-ref-1)
2. IBM Cloud Education: SOA (service-oriented architecture),   
   https://www.ibm.com/cloud/learn/soa [↑](#footnote-ref-2)
3. Bashioum C, Behera P, Breininger K, et al (2006) Reference model for service oriented architecture. OASIS Standard soa-rm, Orga- nization for the Advancement of Structured Information (OASIS), https://docs.oasis-open.org/soa-rm/v1.0/soa-rm.html, version 1.0, accessed 2021-10-05 [↑](#footnote-ref-3)
4. Edwards WK (1999) Core Jini with Book. Prentice Hall Profes- sional Technical Reference [↑](#footnote-ref-4)
5. https://www.fiware.org [↑](#footnote-ref-5)
6. https://www.eclipse.org/basyx/ [↑](#footnote-ref-6)
7. Delsing J (2017) IoT autoamtion - arrowhead framework. CRC   
   Press, Boca Raton [↑](#footnote-ref-7)
8. https://projects.eclipse.org/projects/iot.arrowhead [↑](#footnote-ref-8)
9. Paniagua C, Delsing J (2020) Industrial frameworks for internet of things: a survey. *IEEE Syst J* pp 1–11 https://doi.org/10.1109/ JSYST.2020.2993323 [↑](#footnote-ref-9)
10. https://en.wikipedia.org/wiki/Service-oriented\_architecture [↑](#footnote-ref-10)
11. Adolphs P, Bedenbender H, Dirzus D, Ehlich M, Epple U, Hankel

    M (2015) Referenzarchitekturmodell industrie 4.0 (rami 4.0) [↑](#footnote-ref-11)
12. https://www.iiconsortium.org/IIRA.htm [↑](#footnote-ref-12)
13. Maier MW (1996) Architecting principles for systems-of-systems. INCOSE Int Symp 6(1):565–573. https://doi.org/10.1002/j.2334- 5837.1996.tb02054.x [↑](#footnote-ref-13)
14. Maier MW (1998) Architecting principles for systems-of-systems. Systems Eng 1(4):267–284 https://doi.org/10.1002/(SICI)1520- 6858(1998)1:4< 267::AID- SYS3> 3.0.CO;2- D [↑](#footnote-ref-14)
15. Boardman J, Sauser B (2006) System of systems - the meaning of of. In: 2006 IEEE/SMC international conference on system of systems engineering. p 6 https://doi.org/10.1109/SYSOSE.2006. 1652284 [↑](#footnote-ref-15)
16. Sauser B, Boardman J, Verma D (2010) Systomics: Toward a biol-

    ogy of system of systems. IEEE Trans Syst Man Cybern Part A Syst Hum 40(4):803–814. https://doi.org/10.1109/TSMCA.2010. 2048024 [↑](#footnote-ref-16)
17. Delsing, J., Kulcsár, G. & Haugen, Ø. SysML modeling of service-oriented system-of-systems. Innovations Syst Softw Eng (2022). https://doi.org/10.1007/s11334-022-00455-5 [↑](#footnote-ref-17)
18. J. Delsing, J. Eliasson, J. van Deventer, H. Derhamy and P. Varga, "Enabling IoT automation using local clouds," *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, Reston, VA, USA, 2016, pp. 502-507, doi: 10.1109/WF-IoT.2016.7845474. [↑](#footnote-ref-18)