

Definitions

Site Navigation: [Technical Documentation](#) | [Introduction to EdgeX Foundry](#) | [EdgeX Foundry Microservices Architecture](#) | [Definitions](#)

Click on the following letters for navigation within this table of definitions.

[C - E](#) [G - L](#) [M - O](#) [P - Q](#) [R - Z](#)

Term	Definition
Agnostic	The word <i>agnostic</i> comes from the Greek <i>a-</i>, meaning <i>without</i> and <i>gn?sis</i>, meaning <i>knowledge</i>. Agnostic means "without knowledge." In technology, agnostic means that the item is not specific to a set method or other item, but instead can function with a variety of methods or other items. The agnostic item functions without "knowing" the details of the items with which it is working, or the system with which it is within. Technology agnosticism is accomplished through compliance with widely-used standards or the item is designed and built to be able to interact with a variety of other types of items. Software, including databases, operating systems, platforms; hardware, including devices; standards, protocols, even business processes can be "agnostic." Examples are as follows: • Cross-platform software applications are platform-agnostic and operate on any combination of operating systems and architecture. • Software applications may be device-agnostic and operate on various types of computers, tablets, and mobile devices. • Software applications such as business analytics (BA) and enterprise resource planning (ERP), may be database-agnostic and operate with any vendor's database management system (DBMS). • Software may be protocol-agnostic and be independent of communication protocols by negotiating protocols with its peer, then starting communication. • Software may be business process agnostic with the logic being isolated and the software is able to function in different business processes. • Software, middle ware, hardware, can be vendor agnostic and able to interact with products of different vendors. • Licensing of software may be hardware agnostic and be per device or per user, rather than each license set for a specific device or virtual machine. • Hardware agnostic is also referred to as device independent.
API	<i>See Application Programming Interface</i>
Application Programming Interface	APIs are sets of sub routine definitions, protocols, and tools for creating software applications that often communicate between other software applications and components. APIs are used for a web-based systems, operating systems, database systems, computer hardware, and software libraries, and can include specifications for routines, data structures, object classes, variables, or remote calls.
CEP	Complex Event Processing
Cloud	Cloud refers to cloud computing in which users access applications, data, data storage, or all of those on shared resources, for free or for payment, thereby avoiding the higher cost of installing, owning, and maintaining any, or all of those resources themselves.
Container	Containers are virtual environments for deploying software applications that are isolated from each other but may share Operating Systems, binaries, and libraries. Containers are lighter weight virtual environments with less isolation. Virtual machines or virtual systems have their own set of resources allocated to them, but containers do not. Containers are often adopted by microservice applications where each microservice runs in its own container. EdgeX Foundry can currently be deployed in Docker containers and Snappy containers.
Created time stamp	The Created time stamp is the time the data was created in the database and is unchangeable. The Origin time stamp is the time the data is created on the device, device services, sensor, or object that collected the data before the data was sent to EdgeX Foundry and the database. Usually, the Origin and Created time stamps are the same, or very close to being the same. On occasion the sensor may be a long way from the gateway or even in a different time zone, and the Origin and Created time stamps may be quite different.
DELETE	HTTP method (similar to a command) used in a RESTful API. "DELETE" deletes items. DELETE should be used cautiously and deliberately because it permanently removes items. • For a collection of items such as http://example.com/groupofitems DELETE deletes the ENTIRE collection. • For an individual item such as http://example.com/groupofitems/item92 DELETE deletes just the single item, in this example numbered "92."
Docker	Open source software that automates the deployment of applications inside of software containers. Paraphrased from the website of Docker at www.docker.com: Docker packages application dependencies in containers for portability and redictability for development, testing, and deployment. Docker isolates applications in containers to eliminate conflicts and enhance security. Developers don't have to install and configure complex databases nor worry about switching between incompatible language toolchain versions....that complexity is pushed into containers that are easily built, shared and run. Docker works with any stack and Developers can deploy both microservices and traditional applications.

EdgeX Foundry	Open source software where the Internet of Things meets the Cloud. EdgeX Foundry is supported by Dell.
Fuse	Dell's IoT edge platform that was started in the summer of 2015. Fuse was contributed to the EdgeX Foundry open source community and the Linux Foundation in May 2017. The original Fuse project served as the initial code base for the EdgeX Foundry edge software platform.
GET	HTTP method (similar to a command) used in a Definitions#RESTful API. "GET" retrieves information on items. GET does not alter the item or the data in any way and is "safe" to use. For a collection of items such as http://example.com/groupofitems GET lists the URIs and other details of the items in the collection. For an individual item such as http://example.com/groupofitems/item92 GET shows details of that specific item, in this instance, Item # 92.
HTTP	<i>See Hypertext Transfer Protocol</i>
HW	Hardware
Hypertext Transfer Protocol	An application protocol for distributed, collaborative, and hypermedia information systems. HTTP is the foundation of data communication of the World Wide Web.
Internet of Things	The Inter-networking of physical devices embedded with electronics, sensors, actuators, software, network connectivity, and the software on the physical devices enables the physical devices to provide data to the Cloud through EdgeX Foundry, exchange data among the physical devices, and receive instructions to perform specific actions. IoT physical devices include but are not limited to: appliances in homes, alarm systems, heating and air conditioning, lights, machines in any industry, irrigation systems, drones.
IoT	<i>See Internet of Things</i>
IIoT	Industrial Internet of Things
Java	An object-oriented, high-level computer programming language. Some of EdgeX Foundry is written in Java. Java is different than JavaScript.
Message Queue Telemetry Transport	ISO standard (ISO/IEC PRF 20922) and is a "lightweight" messaging protocol for use on top of the TCP/IP protocol. Message Queue Telemetry Transport (MQTT) is used for connections with remote locations where a "small code footprint" is needed or the network bandwidth is limited.
Metadata	Data that provides information about other data. An example is a catalog of book titles. The catalog itself does not have the content you want to read, but provides information about how to obtain the book that has the content you want to read.
Microservices	Independently developed, deployed, and updated software application components. Individual microservices rarely satisfy an end user's requirements, but a collection of microservices form an application or system that can meet the user's needs. Microservices make changing and adding functionality to a system, easier. The diagram illustrates two architectural models. On the left, 'Monolith Architecture' shows a single large rectangle containing three distinct shapes: a gray square, a blue triangle, and a blue circle. On the right, 'Micro Service Architecture' shows three separate, smaller rectangles, each containing one of the shapes: the gray square, the blue triangle, and the blue circle. Below the diagrams, a legend shows a small square box followed by the text 'Process', indicating that each box represents an independent process.
Microservice Architecture	Method of developing microservices, small, independent, but interacting modules, to interact to solve a problem, or to meet a business goal.
MQTT	<i>See Message Queue Telemetry Transport</i>

MSA	<i>See Microservice Architecture</i>
Nanoservice	Smaller segment than a microservice. Very fine-grained service where the service's overhead outweighs its utility. EdgeX Foundry uses microservices and does NOT use nanoservices.
North Side or North Edge	The "Cloud," where data is collected, stored, aggregated, analyzed, and turned into information is referred to as the "north side" of the network. At the north edge the data collection and enterprise side, EdgeX Foundry enables north side clients to register for the data of interest, dictate its format (for example, JSON, XML), and indicate where and how the data is to be delivered by EdgeX Foundry (for example, to an enterprise REST address or through an MQTT message destined for a cloud based message topic).
Origin time stamp	The Origin time stamp is the time the data is created on the device, device services, sensor, or object that collected the data before the data is sent to EdgeX Foundry and the database. The Created time stamp is the time the data was created in the database. Usually, the Origin and Created time stamps are the same or very close to the same. On occasion the sensor may be a long way from the gateway or even in a different time zone, and the Origin and Created time stamps may be quite different.
PaaS	<i>See Platform as a Service</i>
Platform as a Service	A cloud-based environment in which to develop, run, test, and maintain web-based applications. The PaaS environment provides the entire IT resource stack as a service and provides all of the facilities required to support the complete life cycle of building and delivering web-based applications. Using PaaS enables the creation and deployment of web-based application software without the cost and complexity of buying and managing underlying hardware, operating software, and utilities. Using PaaS therefore enables Developers to accelerate the pace of developing applications while reducing the complexity because Developers can provision, deploy, and manage applications using one unified management system. By using PaaS, Developers get their application to market faster, and can innovate and experiment with new technologies using existing public, private, or hybrid cloud infrastructures.
PLC	Programmable Logic Controller
Polyglot (and Polyglotism)	An application or system written in multiple programming languages and/or using multiple application development frameworks and tools. EdgeX Foundry is a polyglot system because each of the microsystems are written in different programming languages, using many different tools and technologies. To date, EdgeX Foundry microservices are written in Java, Python, Node.js (JavaScript), and Go.
POST	HTTP method (similar to a command) used in a Definitions#RESTful API. "POST" creates or adds new information, a new record or file of items. POST does not alter existing items or existing data in any way and is "safe" to use. For a collection of items such as http://example.com/groupofitems POST creates 1 new item within the collection. For an individual item such as http://example.com/groupofitems/item92 POST is not usually used, since POST creates 1 new item in the collection. New information or a new record, does not have size limitations or limitations for the amount of characters used in the information.
PUT	HTTP method (similar to a command) used in a Definitions#RESTful API. "PUT" replaces or updates information on items. PUT should be used cautiously and deliberately because it permanently removes old data and replaces it with new data. For a collection of items such as http://example.com/groupofitems PUT replaces the entire collection with a new collection. For an individual item such as http://example.com/groupofitems/item92 PUT replaces the specific item of #92 of the collection with new information, or if it that item does not exist, then PUT creates that item and adds it. New information or a new record, does not have size limitations or limitations for the amount of characters used in the information
REST	short for RESTful
RESTful	Representational State Transfer, or RESTful web services. Use REST with EdgeX Foundry. Unlike SOAP-based Web services, no "official" standard for RESTful Web APIs exists. REST is not a standard, but RESTful implementations make use of standards, such as HTTP, URI, JSON, and XML. REST is an architectural style, while SOAP is a protocol. Use 1 of 4 HTTP methods in a RESTful API (see each entry in this table for details on each HTTP method): • GET retrieves information • POST creates/adds new information • PUT replaces/updates information • DELETE deletes/removes information

Rules Engine	Rules engines are important to the IoT edge system. A rules engine is a software system that is connected to a collection of data (either database or data stream). The rules engine examines various elements of the data and monitors the data, and then triggers some action based on the results of the monitoring of the data it. A rules engine is a collection of "If-Then" conditional statements. The "If" informs the rules engine what data to look at and what ranges or values of data must match in order to trigger the "Then" part of the statement, which then informs the rules engine what action to take or what external resource to call on, when the data is a match to the "If" statement. Most rules engines can be dynamically programmed meaning that new "If-Then" statements or rules, can be provided while the engine is running. The rules are often defined by some type of rule language with simple syntax to enable non-Developers to provide the new rules. Rules engines are one of the simplest forms of "edge analytics" provided in IoT systems. Rules engines enable data picked up by IoT sensors to be monitored and acted upon (actuated). Typically, the actuation is accomplished on another IoT device or sensor. For example, a temperature sensor in an equipment enclosure may be monitored by a rules engine to detect when the temperature is getting too warm (or too cold) for safe or optimum operation of the equipment. The rules engine, upon detecting temperatures outside of the acceptable range, shuts off the equipment in the enclosure.
SaaS	<i>See Software as a Service</i>
SDK	<i>See Software Development Kit</i>
Service - Oriented Architecture	A type of software architecture that is a collection of services (such as self-contained, small units of business logic with specific outcomes) that communicate with each other through a communication protocol over a network. The communication can be basic data passing, or complex such as two or more services coordinating an activity.
Snappy	A Linux-based Operating System. The packages are called 'snaps' and the tool for using them 'snapd', and works for phone, cloud, internet of things, and desktop computers. The "Snap" packages are self-contained and have no dependency on external stores. "Snaps" can be used to create command line tools, background services, and desktop applications.
SOA	<i>See Service-Oriented Architecture</i>
Software as a Service	Software as a service (SaaS) is part of cloud computing, and is a software delivery method in which software is licensed by subscription, is cloud hosted, and accessed by the user on the Internet through a web browser. Any software of any kind can be delivered as SaaS.
Software Development Kit	A set of software development tools that enables the creating, the development, of particular software. The software development tools may include application programming interfaces (APIs), libraries, debugging facilities, software utilities, sample code, supporting documentation, hardware, and embedded systems.
South Side or South Edge	All IoT objects, the physical realm, is referred to as the "south side" of the network. The south side is the edge of the network that communicates directly with those devices, sensors, actuators, and other IoT objects. At the south edge the edge of the IoT objects, is a collection of meta-data driven device microservices that communicate in the protocols of choice of the IoT object, and translate sensor data into a EdgeX Foundry common data structure. The microservices also translate EdgeX Foundry actuation requests into physical commands on the device or sensor.
SW	Software
UI	User Interface

[Return to TOP](#)

Site Navigation: [Technical Documentation](#) | [Introduction to EdgeX Foundry](#) | [EdgeX Foundry Microservices Architecture](#) | [API Reference](#) | [Definitions](#)

Where to Next