ActionVector IoT

THE SMART IOT PLATFORM

ARCHITECTURE OVERVIEW

Table of Contents

Introduction

This document describes the architecture and design of the IoT platform that can be used to develop IoT solutions. The framework has been successfully used to develop two reference solutions namely Monitoring and Analytics Solutions for Solar energy and IT Infrastructure domains. This is a work-in-progress document.

This document is targeted for Solution architects, Designers and developers who are keen on developing IoT solutions based on ActionVector IoT Platform.

The reader is advised to read the IoT Platform Terms and definitions document before reading this document.

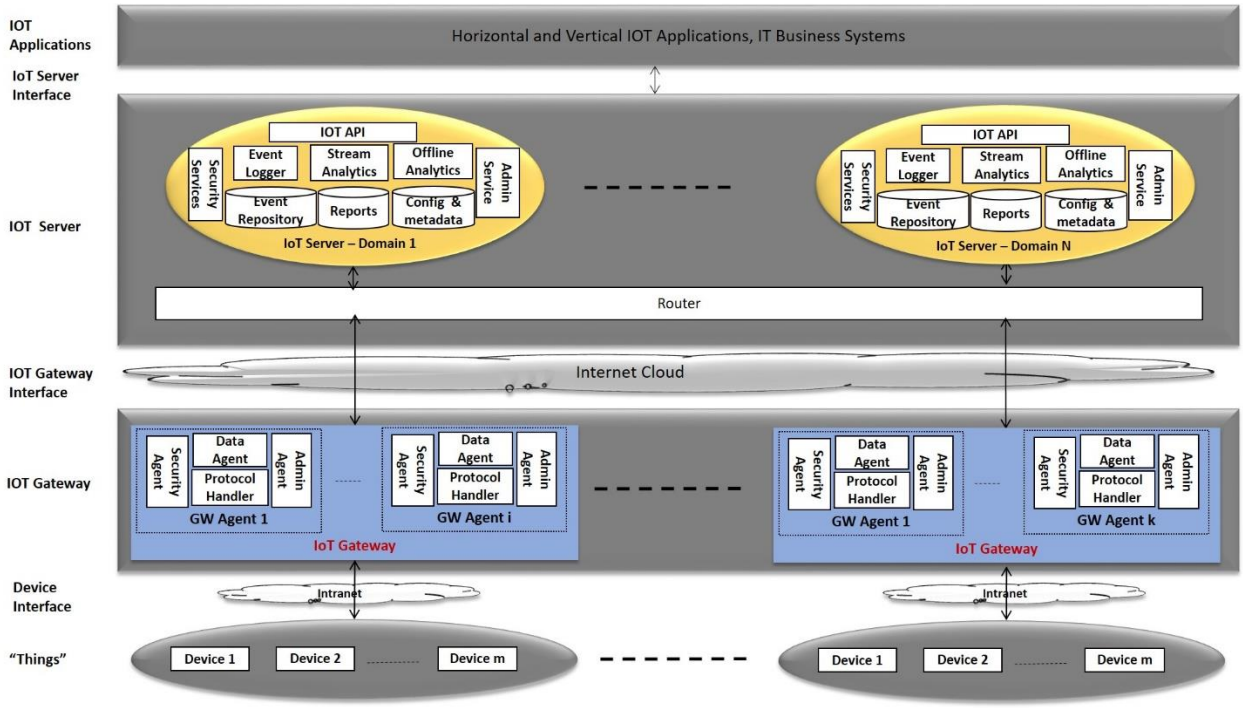
IoT Platform Architecture

There architecture addresses the separation of concerns along two primary dimensions

- The separation of logical or functional tiers comprising of devices, IoT Gateway, IoT Server and application tiers each encapsulating associated functionality.
- The Separation of functional components in each tier based on their affinity to a specific domain of application such as home automation, smart energy, smart city, etc.

The Functional Tiers View

The diagram below shows the overview of the platform architecture in terms of functional tiers with components that encapsulate key functionality in each tier..



"Things"

The lower most tier represents a network of "things" (IoT). A "thing" is any device such as an energy meter, a solar inverter, a computer, a car, a motor, a refrigerator, or even a software component or "anything" that can be controlled, that can respond to commands, that can share its own state information with external systems via some form of communication interface. This could even be a hub of "things" that aggregates data from many devices. A device can be a sensor or an actuator. A logical grouping of devices enables ease of administration and is the basis for multi instance or multitenancy deployment. Each such group of devices is controlled and managed by an IoT Gateway.

The IoT Platform helps build useful real life applications around a network of such intelligent "things" for home automation, energy management, traffic management and many more.

Device Interface

The IoT Gateway interacts with the device (or hub) through well-defined sensor and actuator type of interfaces. The interface may take the form of standards based or proprietary communication hardware interface and communication protocols. Ethernet, wifi, RS232, RS485 are example of typical communication interfaces with communication protocols such as Blue tooth, Modbus, TCP, HTTP, Zigbee etc.

IoT Gateway

An IoT Gateway is a computer such as a general purpose desktop computer, a laptop or a small computer such as Raspberry pi or any computing environment that provides a secure execution environment for Gateway Agent software components that enables secure interactions between the devices and the IOT server on the cloud.

A Data Agent is a key component in the IoT gateway that is responsible for two-way communication between the devices and IoT server. It collects the data from sensor devices on the local intranet via the native interface on one side, and streams the data over the Internet to the IOT cloud server on the other side. Similarly it can trigger the device actuator in response to commands from the IoT Server.

The protocol handler encapsulates the protocol implementation for interacting with the device through the communication interface over the local Intranet. The Security Agents ensure user, storage and communication level security functions. Admin Agents provide device metadata and configuration and exposes interfaces for remote management of devices from IoT server.

One or more Gateway Agent software instances may be deployed in a Gateway to handle multiple different devices of same or different types.

IoT Gateway Interface

The IoT Gateway Interface is a bi-directional interface between the IoT Gateway and IoT Server over the Internet cloud. The REST and TCP are the standard protocols used for communication. The IoT Server communicates with the IoT Gateway for device management as well as control and commands to the devices and IoT Gateway sends to the server, the data collected from the device as well as updates or changes to the local configurations.

IoT Cloud Server

IoT Cloud server is a centralized cloud (private or public) based secure infrastructure comprising of servers running various software components. The software components can be deployed to operate in multi tenancy or multi-instance mode enabling logically or physically separate instances for each domain. The event logger receives and logs the time stamped data from IoT gateway into a database. The Stream analytics enables analytics on data streams in real time whereas the offline analytics component provides analytics and multi-dimensional reports in batch mode. The security services provide user, storage and communication level security functions. The admin services provide metadata, configuration and administration of the domains. The router routes the data to appropriate domains based on the domain identifier contained in the messages to/from the IoT Gateway.

IoT Server Interface

The IoT server exposes well defined set of two-way APIs for external applications to access the data, send commands and receive asynchronous alerts and notifications. The application provide a REST endpoint for receiving alerts and notifications and access the REST API provided by the server for all interactions.

IoT Applications

IoT Applications are domain specific, horizontal or vertical applications. The Application services bring together the aggregated capabilities of a network of devices and enable remote management of the network of devices. The application may be cross-domain or local to a single domain. These applications encapsulate business logic or business rules to interpret the data from the devices and initiate appropriate actions. A ticketing workflow is a good example of a horizontal applications that analyzes the data from various devices and initiates manual workflows for handling service disruptions, performance issues etc.. Similarly, in a solar energy domain for example, a vertical application consumes the data from various solar installations from different regions to analyze the region-wise energy generation, identify the gaps, and root cause analysis and initiate suitable actions or, the energy generation or consumption data received from IoT gateways may be used to trigger finance and accounting transactions in an ERP system.

Similarly, an application service for a smart home domain, enables family members to remotely manage and control the home appliances such as refrigerators, microwave ovens, light bulbs, etc. through their mobile phones. An application service for smart solar energy domain for example, can help remotely monitor and control solar plants comprising of solar panels, inverters, energy meters etc. The Application services may also enable collaboration between device services such as interaction between (say) IoT enabled traffic signals system and IoT enabled vehicles in traffic management in a smart city project. The IoT Platform – the focus of current discussion, is a software platform that enables quick development of such application services in a network of such devices.

The IoT Applications interact with the IoT Server components via public APIs. The IoT applications may be a part of the IoT server ecosystem interacting via loosely coupled generic infrastructure such as Enterprise Service Bus (ESB) or any other proprietary interface.

The IoT platform supports multi-instance or multitenant mode of deployment where different Solutions share a set of common services.

The Domain Affinity View

The motivation for separation of logical tiers in the platform architecture is rather obvious and needs no further explanation. The separation of the functional components in each tier based on their affinity to the domain of application is equally critical. Some components are domain agnostic and some components are sensitive to the domain of application. A "Domain" represents a specific IoT solution domain such as smart homes, smart cities, logistics, smart health, smart energy, retail, etc. It represents an area of business or an industry that an IoT solution is being designed for.

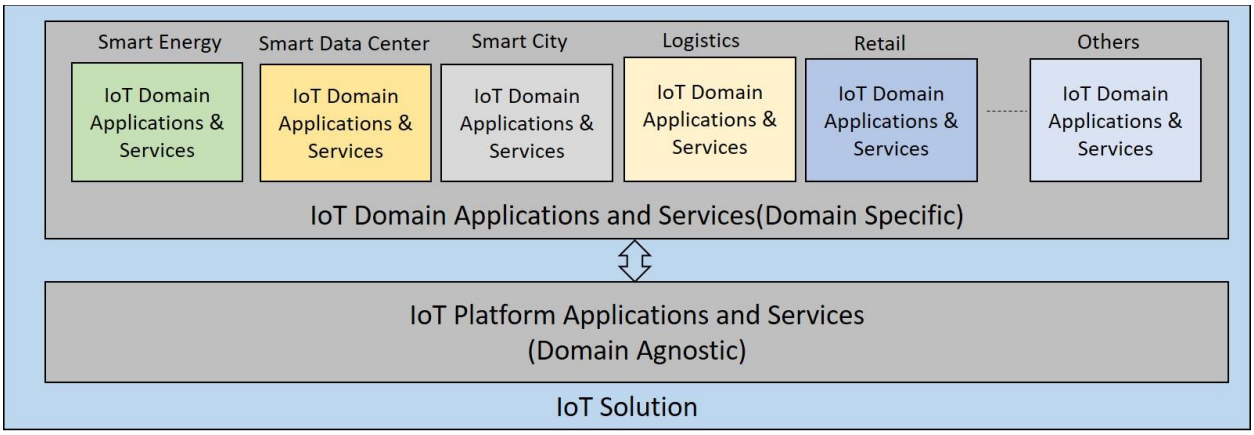
One of the objectives of the architecture is to ensure that the IoT platform architecture is not bound to a specific domain but enables efficient development and deployment of solutions in one or more application domains. The idea is to ensure maximum reusability of domain agnostic and domain specific components.

In the device tier, the type of devices (sensors, actuators), their communication interfaces and protocols, the type of commands, syntax and semantics of the device data can be very different between devices and vendors in a given domain or across domains. The devices may support standards based or proprietary communication interfaces. There may not be a single accepted standard in a given domain. For example, a Modbus protocol is very commonly supported protocols by many electrical and electronics devices cutting across industry domains such as solar devices (such as inverters, meters, etc.), and can be reused across IoT Applications in those domains. On the other hand, there are good number of vendors supporting alternate standards (e.g canbus) or even proprietary interfaces.

Similarly, in the server tier, a Complex Event Processing service is a pure technology service that is not specific to any domain at the same time a set of rules required to detect anomalies, patterns, correlations in the event stream in a given domain can be very specific to the domain.

A closer look at all functional components in the IoT Gateway and IoT Server across diverse domains such as smart city, smart homes, logistics, smart energy etc. will reveal the amount of diversity at both functional and technical levels that the IoT platform needs to handle. Hence the platform architecture needs to clearly provide a way of handling domain specific and domain agnostic aspects of various components in the system in a graceful and efficient manner.

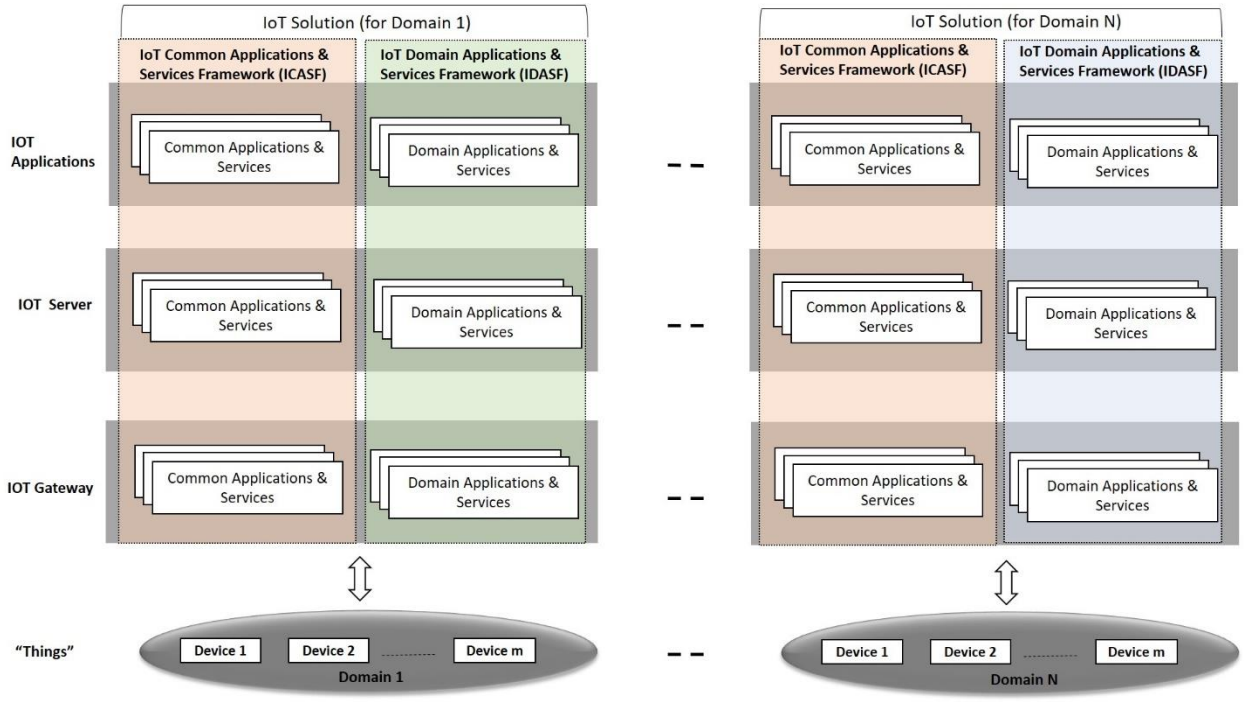
The diagram below shows the logical tiers showing the domain agnostic and domain specific tiers with applications and Services.



The IoT platform applications and services are the domain agnostic applications and services that is the foundation for developing IoT Solution for any domain. This is the IoT platform or the base platform. The IoT Domain Applications and Services are the domain specific applications and Services that are built on the base platform. An IoT solution for a specific domain comprises of base platform (domain agnostic Applications and Services) and the domain specific applications and services.

IoT Platform Architecture View

Combining the logical tier view and the domain affinity view, we arrive a high level architectural view of the IoT Platform as shown below.



The IoT Common Application and Services Framework (ICASF) is the base IoT platform consisting of applications and services that are essential for any IoT Solution. For example, the stream analytics engine, offline analytics framework, event loggers are components that are required for any IoT solution. A domain agnostic services contains no code that is specific to any particular domain.

The IoT Domain Applications and Services Framework (IDASF) on the other had comprise of applications and services that are specific to the IoT application domain such as smart homes, smart energy, supply chain etc. For example, the event stream, a set of rules to filter, aggregate, detect patterns on an evet stream, the reports are specific to an application domain. These services are designed to be reusable across other applicable domains. The domain specific services may include part or all of the code that is specific to a given domain or a set of domains (not common across domains)

The diagram also shows IoT Solutions for different domains comprising of domain agnostics and domain specific frameworks along the logical tiers. IoT solutions for one or more domains can be deployed on the IoT platform. It is this clear separation of domain agnostics and domain specific services across the logical

tiers that is the underpinning of this architecture. A 'Domain' in this context refers to an industry or a business domain of IoT Application.

An IoT application may be a horizontal class of applications such as device monitoring & control applications, performance dashboards or vertical class of applications such as automatic problem detection and ticketing application or a support applications or, it can be a vertical business application such as billing application that generates bills based on the electricity generated by a solar device or integrate into a ERP system in an enterprise and so on. An IoT application may also be a cross-domain application.

The overall architecture can be visualized as a service matrix with each horizontal row representing a logical tier in the system and each vertical column representing the level of domain affinity. The applications and Services in each intersecting cell correspond to the intersecting logical tier and domain affinity.

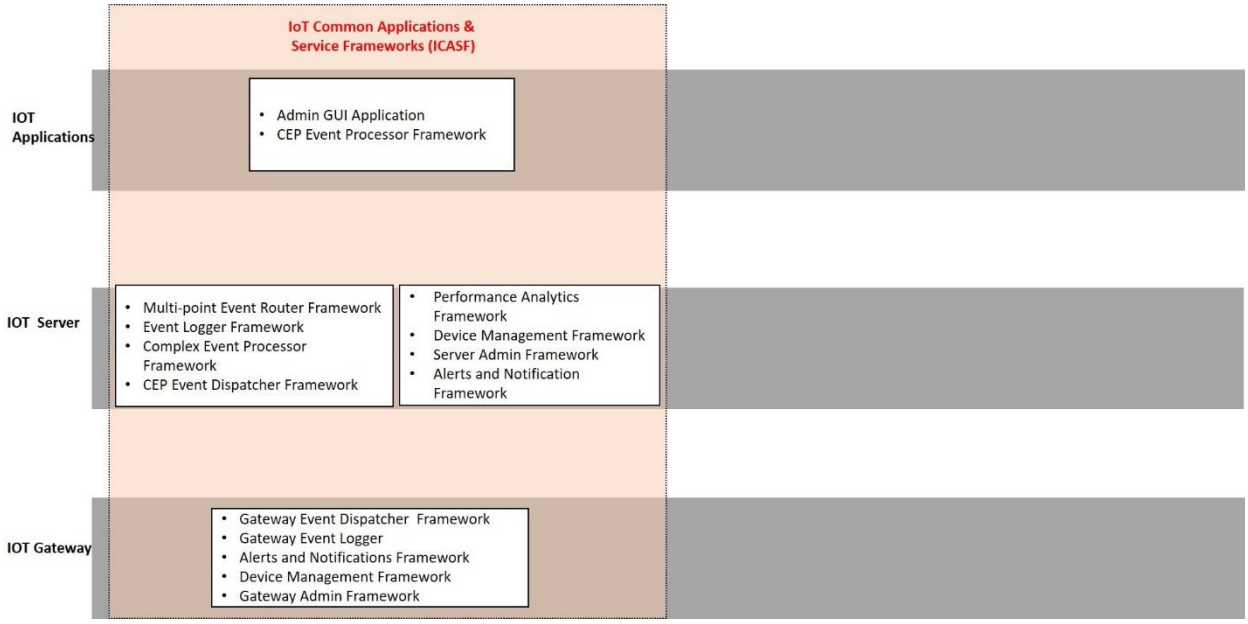
The IoT platform architecture follows the principles of loosely coupled, Service/Micro Service Oriented Frameworks. Each Service Framework is like a building block comprising of one or more micro services, APIs, configuration files and databases. Each framework/service can be customized and independently deployed. The platform design identifies and separates different class of services at diffe rent levels of abstraction that makes the platform extensible and adaptable to different IoT domains and development scenarios.

It is also worth noting that managing and evolving ICASF requires teams with Java programming skills with knowledge of technology platforms and tools whereas developing/evolving IDASF requires technical teams with knowledge of working with low level device interfaces and communication protocols in different domains closely working with domain/business analysts. The Domain specific application development requires application development skills closely working with business/domain analysts.

The following sections will describe the ICASF and IDASF in greater detail.

The IoT Common Applications and Services Framework (ICASF)

The diagram below shows the high level frameworks in Common Applications and Services Framework (ICASF). This is the IoT base platform or the foundation for developing an IoT solution for any domain.



This framework includes domain agnostic applications and services in each logical tier of the reference architecture and facilitates quick development of IoT Domain Services and Solutions. This is considered as base platform or foundation platform for developing any IoT solution. This is a generic framework and comprises of technical services such as Complex Event Processing (CEP) services Framework, Event Repository Services Framework, Report generator services, device management services in the server tier, API Services in the application tier and device admin and configuration, event dispatch services in the gateway tier. The developers can download and use this framework to build their own IoT Solutions.

IoT Gateway - Common Services

The IoT Gateway supports a set of common services that encapsulate part of IoT Gateway functionality that is domain agnostic.

Gateway Event Dispatcher Framework

This Framework provides services for dispatching the events from the devices to the IoT Server over TCP or HTTP in a secure manner. This framework also handles dispatching events in batch mode. Batch dispatch is triggered when there is a failure in communication between Gateway and Server. Alternately, the batch file with failed events can also be copied into another computer and directly uploaded to the server via admin application.

Gateway Event Logger Framework

This framework is responsible for logging the time stamped events from the device. The events that could not be dispatched to the server are marked and dispatched by Event Dispatcher in a batch mode.

Alerts and Notification Framework

This framework handles various modes of sending SMS and email alerts and notifications to pre - configured mobile number and email id of the concerned executives. This is invoked when there is a communication failure between Gateway and server or between device and the gateway or any other exceptions conditions occur in the gateway.

Device Management Framework

The Device Management Framework includes services and applications for managing the device setup and configurations locally via local GUI application or remotely from the server via the respective APIs. This includes services for configuring, auto discovery, device setup etc. There is a two-way secure interaction possible between the device management framework on server and IoT gateway side to keep the configurations in sync.

Admin Framework

The Admin Framework handles rest of the general admin functions such as setting up the system level configurations that are not part of the device management framework. Similar to the device management framework, admin framework includes services, APIs and applications for local admin applications and two-way interactions between the Admin Framework on server and IoT Gateway to keep all the configurations in sync.

IoT Server - Common Services

The IoT server supports a set of common services that encapsulate part of IoT Server functionality that is domain agnostic.

Multi-point Event Router Framework

The multi-point Event Router Framework receives device events from IOT Gateway Dispatcher and in turn dispatches the events to one or more configured end points. This Framework should be configured to two mandatory service end points namely Event Logger Framework and Complex Event Processor Framework. This may be configured to deliver the events to additional service end points.

Event Logger Framework

This framework logs the server events into a database. The events may be further filtered, transformed, aggregated before saving into the database. This database is the master repository of the events that is used by the Analytics Framework for performance reports. The time stamped, transformed events are stored along with the SLA/OLA thresholds values used to check for violations. This allows for dynamic thresholding capability.

CEP Framework

Receives real time event stream from the devices and rules are applied on the event stream to identify correlations and detecting anomalies and publish the same to a configured service end point.
This is based on Esper CEP: https://esper.com/

CEP Event Dispatcher Framework

The CEP events detected after executing the CEP rules (filter, aggregate, pattern detection) are dispatched to a configured service end point (REST) in the application domain. This framework should be configured to the CEP Event Processor Framework which is the service end point in the application domain. Applications will use these events to trigger appropriate actions.

Performance Analytics and Reports Framework

This Framework includes services and APIs to analyze the device performance data captured in the Event Logger database and generate reports. The framework generates statistical summaries such as average, min, max, deviations for various metrics associated with the devices, groups (installation sites)

and the customer along various time dimensions such as hourly, daily, weekly, monthly and yearly periods. It provides APIs for IoT Applications to retrieve this and automatic day end summary generation and dispatch to pre-configured email ids.

Device Management Framework

The server side device management framework provides API for centralized device management applications to manage the information and configuration of devices spread across the geographical regions. This includes the device profiles, specific metric names, their SLAs etc. There is a secure two-way interaction possible between the Gateway side and server side frameworks to upkeep the device information and configuration updates in sync.

Admin Services Framework

The server side Admin framework provides API for centralized admin applications to manage the configuration of various software components on the server and the Gateway. There is a secure two-way interaction possible between the Gateway side and server side frameworks to keep the configurations configuration updates in sync.

Alerts and Notification Services

SMS and E-mail Utility to send alerts and notifications. This service is used in IoT Gateway to send alerts and notifications when an exception occurs. It can be used in any application to send SMS/E-mails.

IoT Application - Common Services

CEP Event Processor Framework

This is the CEP event consumer in the application tier. The end applications in the application tier will retrieve the events from the CEP Event Processor Framework and trigger appropriate actions.

Admin GUI Application

The Admin GUI Application provides device management and other general admin functions via the server side device management and admin APIs. This is a reference application. Developers can modify or develop new applications using this application as a template.

The IoT Domain Applications and Services Framework (IDASF)

This IoT platform includes domain specific applications and services. The IDASF Applications and Services provided in the platform are based on two successful IoT solutions implemented using ActionVector IoT Platform. The two solutions are –

a) Solar Plant Monitoring and Analytics Solution for ***Smart Energy Domain***

b) IT Infrastructure Monitoring and Analytics Solution for a ***Smart Data Center Domain***

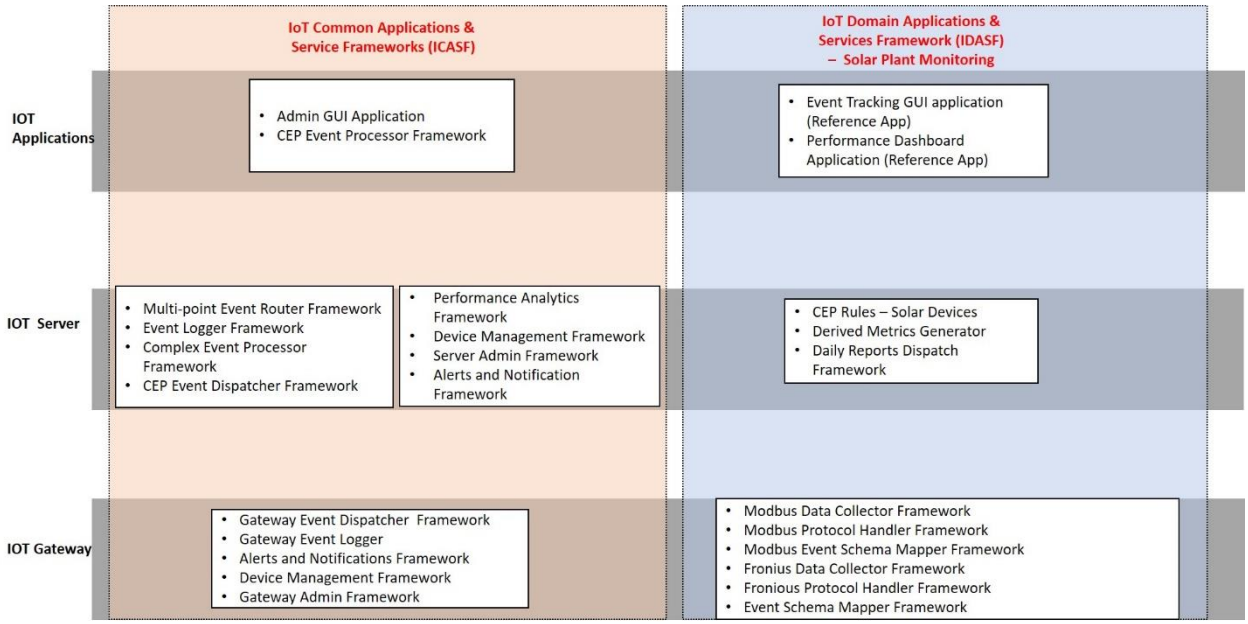
The domain specific applications and services supported in the platform release are reference implementations only. Developers are free to use these applications and services as-is or enhance them to build new solutions in respective domains or build solutions for new domains.

The features and capabilities available in these reference implementation is a subset of the capabilities and features supported in original IoT Solution implementations. Refer to the case study document for details of these solutions.

Following sections will describe the various applications and services in IDASF for the two reference domains - IDASF for Solar Plant Monitoring and Analytics IT Infrastructure Monitoring and Analytics domains.

The IDASF – Solar Plant Monitoring and Analytics

This framework (IDASF – Solar Monitoring) includes domain specific applications and services for Solar Plant Monitoring and Analytics solution. It facilitates quick development of new IoT Plant Monitoring Solutions for Smart Solar energy domain. The diagram below shows the IoT platform with ICASF (discussed in previous section) and IDASF for Solar Plant Monitoring and Analytics Solution.



IoT Gateway – Domain Services

Modbus Data Collector Framework

This Framework behaves like an adapter between the Modbus devices and the server. It interacts with respective protocol handler framework to retrieve the metrics/data from the devices log the events via Event Log Framework and map the device specific metric names and data types to common event schema via mapper framework and finally dispatches the event it to the Server via the Gateway Event Dispatcher Framework.

Modbus Protocol Handler Framework

This framework supports the standard Modbus protocol supported by many solar devices in the market such as solar inverters, meters and other devices. This is used the Modbus Data Collector Framework to interact with the Modbus devices to collect the metrics/data.

Modbus Schema Mapper Framework

This framework encapsulates the logic to map the Modbus device specific metric names and unit of measures (device specific schema) to a common, consistent schema with metric names and Unit of Measures for all further processing by the downstream services.

Fronius Data Collector Framework

This Framework behaves like an adapter between the devices and the server. It interacts with respective protocol handler framework to retrieve the metrics/data from the devices log the events via Event Log Framework and map the device specific metric names and Unit of Measures to common event schema via mapper framework and finally dispatches the event it to the Server via the Gateway Event Dispatcher Framework.

Fronius Protocol Handler Framework

This framework supports the proprietary protocol specific to Fronius devices such as solar inverters, meters and other devices. This is used the Fronius Data Collector Framework to interact with the Fronius devices to collect the metrics/data.

Fronius Schema Mapper Framework

This framework encapsulates the logic to map the Fronius device specific metric names and unit of measures (device specific schema) to a common, consistent schema with metric names and Unit of Measures for all further processing by the downstream services.

IoT Server – Domain Services

CEP Rules

The CEP executes a set of CEP rules on the incoming event streams from Solar devices. The rules are designed to perform various validation checks, aggregations, filtering, correlations, pattern detection on the event stream and these rules constitute the primary domain specific services on the IoT server. The developers may modify existing rules or design new rules as required.

In case of Solar monitoring domain, the following are the two additional services specific to the Solar monitoring domain. Developers can implement more such domain specific services if required. These are good examples that demonstrate the modularity and extensibility of the platform for different domain specific needs.

Derived Metrics Generator

APIs to generate derived metrics. Metrics directly supported in a device are primary metrics such as cpu usage, disk usage in servers or energy generated, voltage, current in a Solar inverter. This service is specific to Solar monitoring domain where the secondary metrics are derived from the primary metrics with custom implementation. The Service gets invoked if configured to generate derived metrics.

Daily Reports Dispatch Framework

This framework automatically generates a summary of the performance of the Solar installations as a Excel spreadsheet and dispatches the report to a list of pre-configured email ids at pre-configured time of the day.

IoT Application – Domain Service

Event Tracker Application

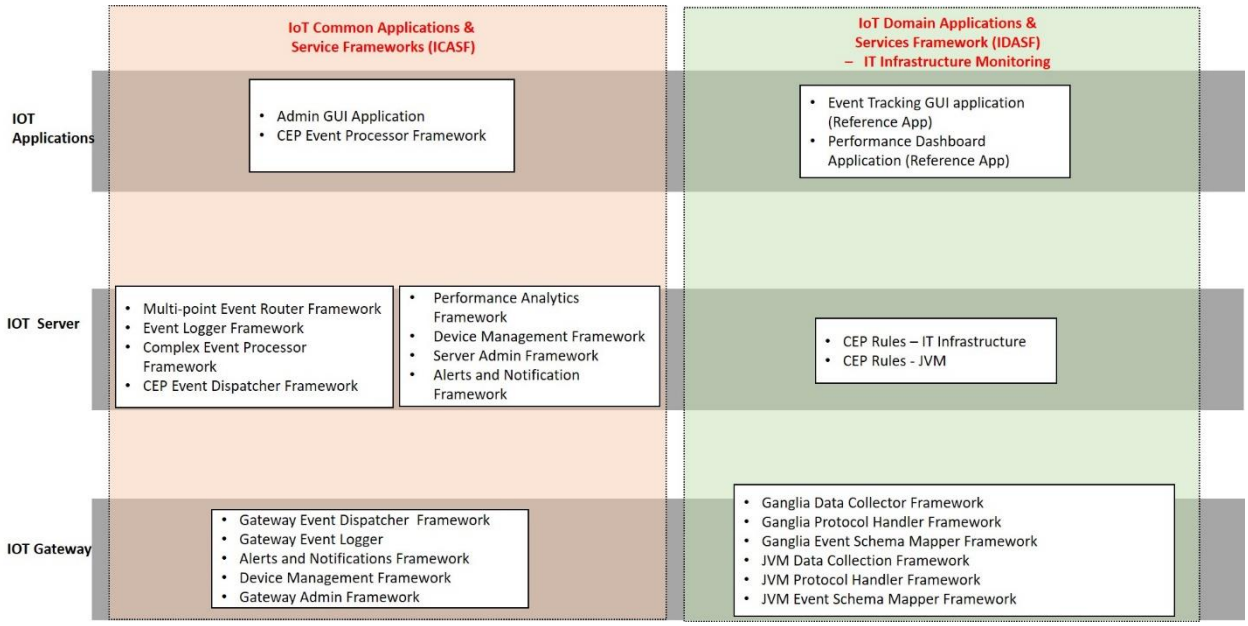
This is a reference application that shows how to retrieve events from the CEP Event Processor Framework. It retrieves the events from the CEP Event Processor Framework and displays the list. Developers can extend this and build their own application logic to process the CEP events.

Performance Dashboard

This is a reference applications that retrieves the device metric values and aggregated chart data from the Performance Analytics and Reports Framework and provides a comprehensive dash board with rich set of user selectable filters. Developers can build their own enhancements or a totally new application using the performance data of various devices in geographically distributed installation sites across one or more customers.

The IDASF – IT Infrastructure Monitoring and Analytics

This framework (IDASF – IT Infra Monitoring) includes domain specific applications and services for IT Infrastructure Monitoring and Analytics Solution. It facilitates quick development of new IoT Solutions for IT Infrastructure domain. The diagram below shows the IoT platform with ICASF (discussed in previous section) and IDASF IT Infrastructure Monitoring and Analytics Solution.



IoT Gateway – Domain Services

Ganglia Data Collector Framework

This Framework behaves like an adapter between the Ganglia nodes (computers monitored by Ganglia) and the IoT server. It interacts with respective protocol handler framework to retrieve the metrics/data from the computers and log the events via Event Log Framework and map the computer specific metric names and data types to common event schema via mapper framework and finally dispatches the event it to the Server via the Gateway Event Dispatcher Framework.

Ganglia Protocol Framework

This framework supports the protocol to send HTTP message to Ganglia Gmetad port and retrieve the XML file comprising of several metrics concerning the health of the computer. This is used by the Ganglia Data Collector Framework to interact with the Ganglia nodes (computers) to collect the metrics/data.

Ganglia Schema Mapper Framework

This framework encapsulates the logic to map the Ganglia specific metric names and unit of measures (device specific schema) to a common, consistent schema with metric names and Unit of Measures for all further processing by the downstream services.

Java Virtual Machine (JVM) Data Collector Framework

This Framework behaves like an adapter between a JVM instance and the IoT server. It interacts with respective protocol handler framework to retrieve the metrics/data from the computers and log the events via Event Log Framework and map the computer specific metric names and data types to common event schema via mapper framework and finally dispatches the event it to the Server via the Gateway Event Dispatcher Framework.

JVM Protocol Framework

This framework supports the protocol to retrieve various metrics from JVM instance using JMX API. This framework is used by the Java Data Collector Framework to interact with the multiple JVM instances (computers) to collect the metrics/data.

JVM Schema Mapper Framework

This framework encapsulates the logic to map the JVM specific metric names and unit of measures (device specific schema) to a common, consistent schema with metric names and Unit of Measures for all further processing by the downstream services.

IoT Server – Domain Services

CEP Rules

The CEP executes a set of CEP rules on the incoming event streams from Ganglia. These rules perform various validation checks, aggregations, filtering, correlations, pattern detection on the event stream and these rules constitute the primary domain specific services on the IoT server. The developers may modify existing rules or design new rules as required.

Please note that IoT server functionality supported for IT Infrastructure monitoring domain agnostics. There are no other domain specific services required in the server. However developers can support new domain specific functionality with appropriate level of customizations.

IoT Application – Domain Services

Event Tracker Application

This is a reference application that shows how to retrieve events from the CEP Event Processor Framework. It retrieves the events from the CEP Event Processor Framework and displays the list. Developers can extend this and build their own application logic to process the CEP events.

Performance Dashboard

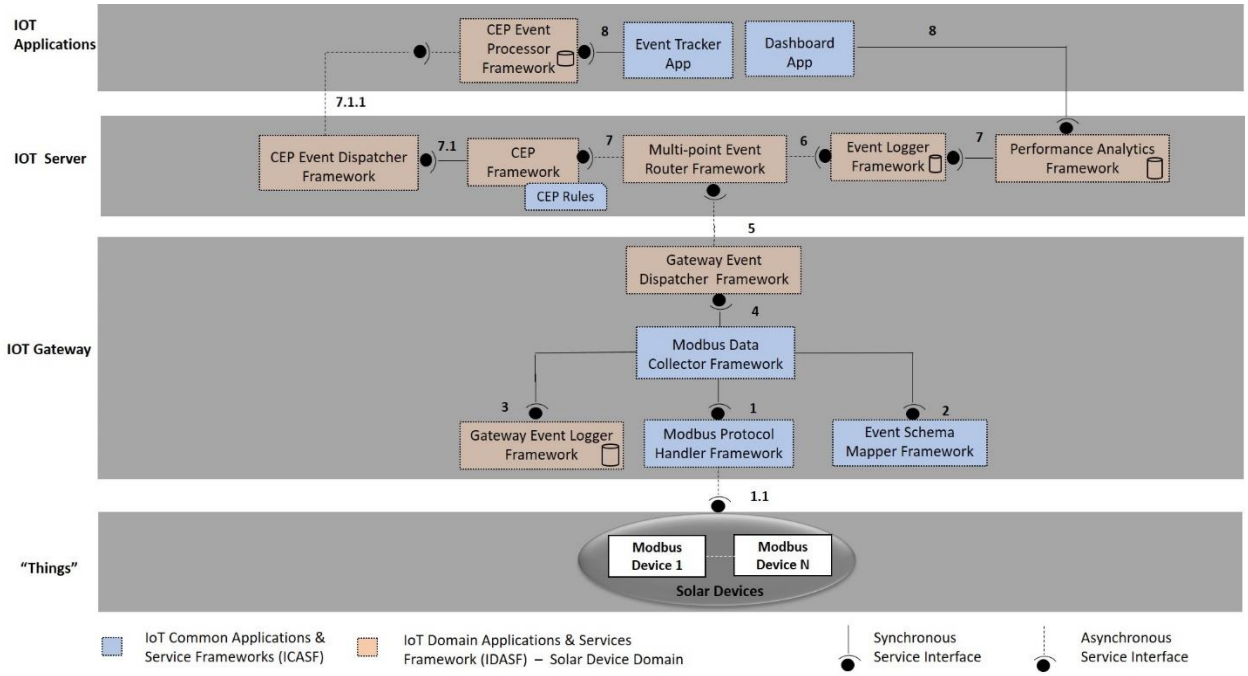
This is a reference applications that retrieves the device metric values and aggregated chart data from the Performance Analytics and Reports Framework and provides a comprehensive dashboard with rich set of user selectable filters. Developers can build their own enhancements or a totally new application using the performance data of various devices in geographically distributed installation sites across one or more customers.

Technology and Platform Stack

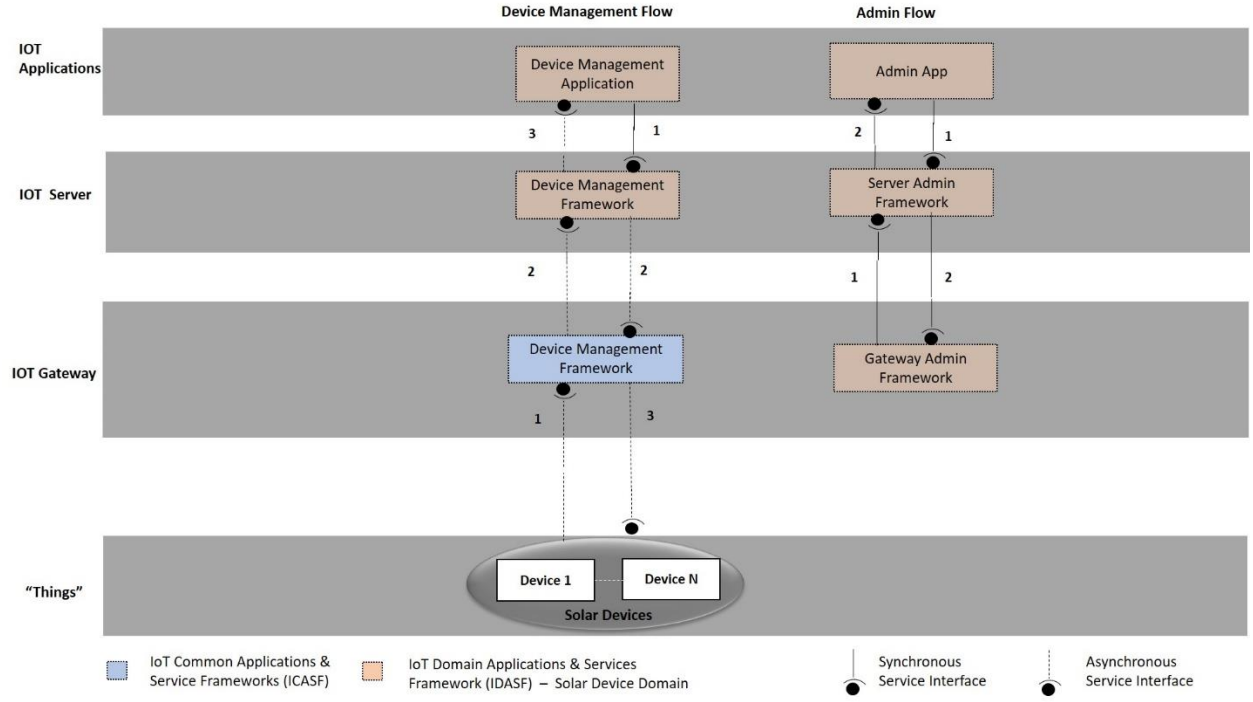
Component	Technology/Platform
Language Platform	Java SE, Java EE
Database	PostGreSQL
Middleware	Tomcat
Server monitoring	Ganglia , OpsView
Stream Analytics	Esper Complex Event Processing Engine (CEP) (Open source)
Ticketing workflow	Prokosha's proprietary BPM Platform
Packaging	Docker

Sample Usecase Flows

Normal Event Processing Flow



Device Management Flow



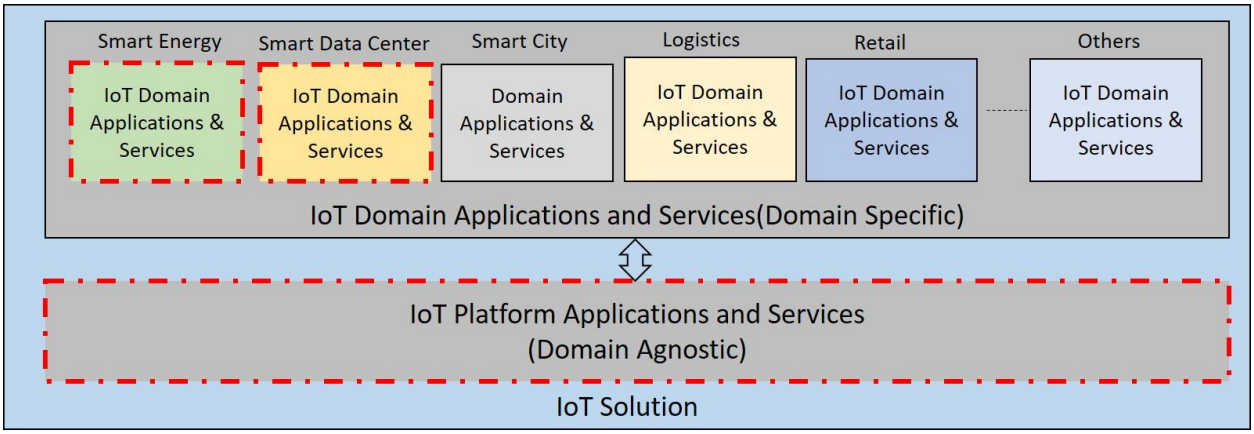
Admin Flow

TBD

IoT New Solution Development and Customization

Solution Development Scenarios

The diagram below shows the IoT platform comprising of domain agnostics platform applications and services and solutions for two reference domains Solar Plant Monitoring and Analytics and IT Data center monitoring and analytics solutions.



Developers have the following options:

a) Solutions for Smart Energy and Smart Data Center - The current version of the platform has a reference implementation of Domain Applications and Services for Smart Energy and Smart Data Center domains. The developer can use this as a base solution and is free to modify, enhance or extend it as required.
b) Solutions for any other domain – The developers have to identify and implement the Domain Applications and Services (IDASF) for the new domain. The applications and services provided in IDASF for Smart Energy and Smart Data Center domains can be used as a template to build IDASF for a new domain

A step-by-step methodology to build new IoT solutions using ActionVector IoT Platform is described in a separate document "IoT Solution Development Methodology" (TBD).

Configuration and Customization Approach

The IoT platform can be visualized as a set of independent service frameworks (building blocks) interacting though well-defined APIs. Each framework is characterized by:

a) Interface (APIs) – Incoming and outgoing APIs. Incoming APIs are APIs for other frameworks to consume its services. Outgoing APIs are APIs of other frameworks consumed by a framework.
b) Implementation – The underlying implementation (services + data) encapsulated in one or more objects
c) Configuration – A set of configuration parameters that determine the runtime behavior of the functionality encapsulated by the framework

The IoT Platform is highly configurable and customizable. There are three options available for controlling or modifying the runtime platform behavior.

a) Configuration (Level 0) - The runtime configurations supported in each framework provides the first level of controlling or modifying the runtime behavior of the platform. This involves setting appropriate values for various configuration parameters in each framework. This involves no changes to the code.
b) Framework Customization (Level 1) – Framework customization is the next level option, If the target requirements demand changes to the underlying implementation of a framework. The customization of a framework typically involves internal changes to the code, databases and configurations of a framework without impacting the APIs. In this scenario, one or more frameworks can be modified.
c) Platform Customization - If the target requirements cannot still be met with the framework customization, the last option is the platform customization. The platform customization may involve changes to the internal services, configurations and databases of one or more frameworks in cluding APIs. For all practical purposes this will be a new IoT platform. It is however, recommendable comply to the architectural principles and guidelines that can cater to another genre of IoT applications.

The configuration and customization of each framework are described in a separate document. However, we identify here high level scenarios that can be handled by configurations and customizations.

Configuration (Level 0)

The various runtime configuration options provided in the platform caters to varying needs of different solutions in different domains. For example, the schema mapper allows mapping the domain specific

device information schema to a common IoT platform schema with configuration. Similarly, the Report generators can be configured to cater to varying needs of different solutions in different domains.

Following scenarios can be handled by configurations in ICASF and IDASF.

1. New device schema or changes to existing device schema
2. New CEP rules on an event stream from a device
3. Setting up of new device profiles, device metrics and thresholds
4. Additional CEP event processor chain for different types of event
5. Configure custom event filters, aggregator, transformers object references (ETL adapter)
6. Configure the metrics used for reports
7. Configure the chart types for different metrics
8. Metrics mapper (Event schema) to user domain
9. New reports/ custom reports

Customization (Level 1)

Customization refers to modifications and enhancements to the internal services, configurations, databases etc. but conforming to the incoming and outgoing APIs.

Following scenarios can be handled by customization of ICASF and IDASF

1. A new device type with a standards based or proprietary communication interface (hardware and software protocols)
2. An existing device type with a different communication protocol
3. An existing device type with different information schema
4. Custom Event Processor (CEP) (stream analytics)
5. CEP event dispatcher to ticketing systems like OTRS, Prokosha BPM Workflow or any custom
6. Alerts and Events notifications – New alerts and notifications, message formats etc.
7. Derived metrics generator/Custom offline analytics processor
8. Custom Event filters and aggregation and transformation objects
9. New reports/ custom reports generator

Platform Customization (Level 2)

No specific scenarios have been identified at this stage. A platform with this level of customization becomes a new custom IoT platform. It is recommended to design the custom IoT platform complying to the architectural principles and guidelines that can cater to another genre of IoT applications.