SHEETS	13 pages	IoT Documentation
PUBLICATION DATE	2018-11-14	

Paraffin
Open Source IoT Platform
Proposal

Ver. 0.1.3

REVISION	DESCRIPTION	BY	DATE	CHECKED
		REVISED		
D3				
D2		H.M		
D1	FOR INFORMATION	H. M	2018-11-14	

	DESCRIPTION	SIGNATURE	DATE	DOC NO.	
DESIGNED					Ver.0.1.2
CHECKED		H.M			
APPROVED		H.M	2018-11-14		

Definition

IoT or The Internet of Things is the network of physical devices, vehicles, home appliances and other items embedded with electronics, software, sensors, actuators, and connectivity which enables these objects to connect and exchange data.

MQTT (Message Queuing Telemetry Transport) is an ISO standard (ISO/IEC PRF 20922) publish-subscribe-based messaging protocol. It works on top of the TCP/IP protocol. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited.

CoAP Constrained Application Protocol (**CoAP**) is a specialized Internet Application Protocol for constrained devices, as defined in RFC 7252. It enables those constrained devices called "nodes" to communicate with the wider Internet using similar protocols. CoAP is designed for use between devices on the same constrained network (e.g., low-power, lossy networks), between devices and general nodes on the Internet, and between devices on different constrained networks both joined by an internet. CoAP is also being used via other mechanisms, such as SMS on mobile communication networks.

Introduction

We are living in the age of connectivity, and that means more things than ever before are right at your fingertips — literally. With one press of the button, one swipe left or right, you can open up new worlds in seconds. The IoT is one of the hottest topics right now, and everyone is trying to figure out how to play in it. Every time someone launches a new product, the market changes and other players in the field need to adapt their offers to the new context.

What is the problem?

Today Startups needs get ready solutions to launch their products.

IoT needs back-end and front-end modules to handle the device connectivity and keep good user experience. The growth of IoT startups and their needs send a pulse to market for running new open source projects with more features like Social IoT and Machine Learning. Today, It is a requirement to share your devices with your friends to use once time or set your home temperature according weather or habit of use.

Scope

The scope of the project is:

- Define and build a social solution to support communication between users and handle sharing devices between users.
- Define and build a security solution to support the communication between all these devices in multi-tenant.
- Define a REST API to expose the machines needs through REST, exposing multiple protocols (MQTT, CoAP) through the same API.
- Define a REST API for admin and users to interact with server and mange devices and settings.
- Prepare a starter kit with capability of building for different platforms so startup holders can be able to launch apps quickly.
- Design an admin and users Dashboard to manage devices and settings.
- Design a Docker/Kubernetes code to build microservice architecture for platform.
- Design a Command Line Interface CLI for users so they can be able to interact with platform. Users can host service on Cloud by payment.
- Design an Installer Script to install all codes automatically.

Description

- Social IoT: a social solution helps to make social IoT network. It should handle social activity like share, Like, Comments on IoT asset.
- Authentication: IoT security needs authentication and authorization service to support the communication security between all of these devices in multi-tenant.

- REST API: multiple protocols (HTTP, MQTT, CoAP) through the same API make it easy for developers to add different features.
- Starter kit for different platform accelerate startups and it helps this platform to be more favor.
- Dashboard help admin and users to manage platform and settings by web.
- Design a Docker/Kubernetes code to build microservice architecture for platform.
- Installer Script and Command Line Interface CLI are tools which can help users to interact with platform without touching the platform complexity. It may encourage users to use built-in features like host service on Cloud by payment.

What won't it do

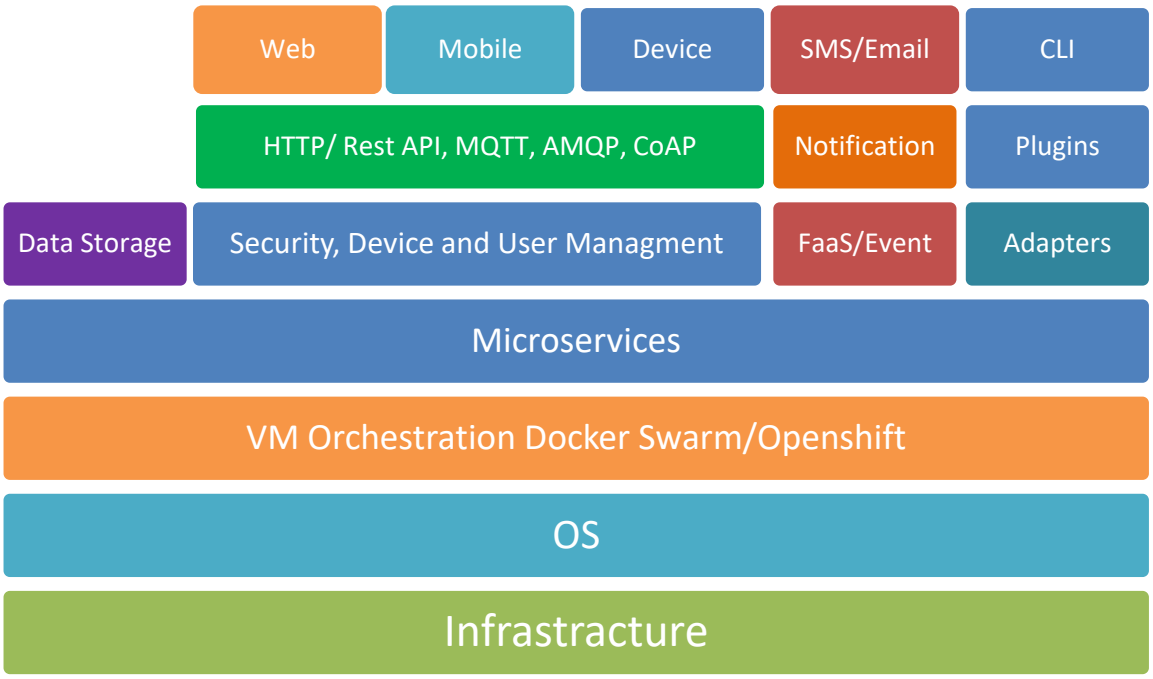
We are not going to reinvent the wheel. This platform is based on opensource codes. the router and broker side code are chosen from proven open source codes with suitable license.

What will it do

All services are related with front-end and management codes will designed. In fact, services like, API Server, Dashboard, Event Manager, Billing and Logging is designed by open source middle ware. Definitely, it must be avoided to rewrite codes which is available in git networks. It is a challenge to find a moderate path so programmer use proper middleware and has not fallen into trap to write codes which can be found open source.

Architecture

The microservices architecture help to tackle the problem of complexity by decomposing application into a set of manageable services which are much faster to develop, and much easier to understand and maintain. It enables each service to be developed independently by a team that is focused on that service.



IoT Architecture

Backend

This platform is based on docker machine. It's necessary to design a proper API for communication between services.

Here there are suggestions for choosing different elements;

- Broker: Mosca, Ponte, ActiveMQ Artemis
- Gateway, Router, Load balancer: Kong for API gateway / Nginx, Traefik, HAProxy / Hono Eclipse for IoT Messaging
- API Server: Parse Server / ExpressJS
- Security, Identity and Access Management IAM: Keycloak
- Function as a Service FaaS: OpenFaas
- Social Activity: Stream Framework
- Container Orchestration: Docker Swarm, Openshift and Kubernetes

Frontend

There are many solutions and frameworks. It worth to effort to work with React or VueJs. Parse Server has good SDKs for HTTP and NodeJS clients. The first target client is Web and the others platforms will be added finally.

Plugins

Users will able to add plugins as adapter to design custom services like Machine Learning. Designing a template for plugin is a big easiness for developers to customize this platform.

API

API Server is based on Parse Server. Parse Server is an open source version of the Parse backend that can be deployed to any infrastructure that can run Node.js. You can find the source on the GitHub repo.

Objects API

URL	HTTP Verb	Functionality
/api/classes/<className>	POST	Creating Objects
/api/classes/<className>/<objectId>	GET	Retrieving Objects
/api/classes/<className>/<objectId>	PUT	Updating Objects
/api/classes/<className>	GET	Queries
/api/classes/<className>/<objectId>	DELETE	Deleting Objects

- http://localhost:1337/api

Users API

URL	HTTP Verb	Functionality
/api/users	POST	Signing Up Linking Users
/api/login	GET	Logging In
/api/logout	POST	Logging Out
/api/users/<objectId>	GET	Retrieving Users
/api/users/me	GET	Validating Session Tokens Retrieving Current User
/api/users/<objectId>	PUT	Updating Users Linking Users Verifying Emails
/api/users	GET	Querying Users
/api/users/<objectId>	DELETE	Deleting Users
/api/requestPasswordReset	POST	Requesting A Password Reset

Sessions API

URL	HTTP Verb	Functionality
/api/sessions	POST	Creating Restricted Sessions
/api/sessions/<objectId>	GET	Retrieving Sessions
/api/sessions/me	GET	Retrieving Current Session
/api/sessions/<objectId>	PUT	Updating Sessions
/api/sessions	GET	Querying Sessions
/api/sessions/<objectId>	DELETE	Deleting Sessions
/api/sessions/me	PUT	Pairing with Installation

Roles API

URL	HTTP Verb	Functionality
/api/roles	POST	Creating Roles
/api/roles/<objectId>	GET	Retrieving Roles
/api/roles/<objectId>	PUT	Updating Roles
/api/roles/<objectId>	DELETE	Deleting Roles

Files API

URL	HTTP Verb	Functionality
/api/files/<fileName>	POST	Uploading Files

Analytics API

URL	HTTP Verb	Functionality
/api/events/AppOpened	POST	Analytics
/api/events/<eventName>	POST	Custom Analytics

Push Notifications API

URL	HTTP Verb	Functionality
/api/push	POST	Push Notifications

Installations API

URL	HTTP Verb	Functionality
/api/installations	POST	Uploading Installation Data
/api/installations/<objectId>	GET	Retrieving Installations
/api/installations/<objectId>	PUT	Updating Installations
/api/installations	GET	Querying Installations
/api/installations/<objectId>	DELETE	Deleting Installations

Cloud Functions API

URL	HTTP Verb	Functionality
/api/functions/<name>	POST	Calling Cloud Functions

/api/jobs/<name>	POST	Triggering Background Jobs

Schemas API

URL	HTTP Verb	Functionality
/api/schemas/	GET	Fetch All Schemas
/api/schemas/<className>	GET	Fetch Schema
/api/schemas/<className>	POST	Create Schema
/api/schemas/<className>	PUT	Modify Schema
/api/schemas/<className>	DELETE	Delete Schema

Function Hooks API

URL	HTTP Verb	Functionality
/api/hooks/functions/<functionName>	GET	Fetch Cloud Functions
/api/hooks/functions/	POST	Create Cloud Function
/api/hooks/functions/<functionName>	PUT	Edit Cloud Function
/api/hooks/functions/<functionName>	DELETE	Delete Cloud Function

Trigger Hooks API

URL	HTTP Verb	Functionality
/api/hooks/triggers/<className>/<triggerName>	GET	Fetch Cloud Trigger
/api/hooks/triggers/	POST	Create Cloud Trigger
/api/hooks/triggers/<className>/<triggerName>	PUT	Edit Cloud Trigger
/api/hooks/triggers/<className>/<triggerName>	DELETE	Delete Cloud Trigger

DB Structure

User

Field Name	Type	Description
objectId	String	unique
createdAt	Date	created at
updatedAt	Date	updated at
ACL	acl	
name	String	unique name like username
email	String	
phone	String	
profile	JSON	

Thing

Field Name	Type	Description
objectId	String	unique
createdAt	Date	created at
updatedAt	Date	updated at
ACL	acl	
threadId	Pointer	point to User
version	String	1.0
type	String	device, app
tenant	String	unique
auth	String	openid , jwt, pask, x509-cert, hashed-password
clientId	String	unique
name	String	like Username

secrets	Array	[
		{
		“adapter”: “mqtt”,
		"pwd-hash": "bFYr5lBhwg=",
		"salt": "vde87=\iDB",
		"hash-function": "sha-512"
		},
		{
		“adapter”: “http”,
		"pwd-hash": "fdjwagtv43fg_",
		"salt": "kkjgf&4wfd",
		"hash-function": "sha-512"
		}
		]

adapter	Array	[‘mqtt’, ‘http’, ‘coap’] or [‘all’]
topics	Array	[‘home, ‘garden/temp’] or [‘*’]
permission	JSON	{‘write: true, ‘read’: true, ‘verifiy’:’always’}
social	-	-
enabled	JSON	{‘not-before’:’2016-05-26T02:07:34.545Z’, ‘not-after’:’2016-05-26T02:07:34.545Z’, ‘enabled’: true}

Timeline

We've offered a basic guide for the timeframe of delivery. It is estimated that it will take 6 months to complete codes and tests for the first stable version. Here's what to expect:

Design	Development	Test	DevOps	Release

Item	1st	2nd	3rd	4th	5th	6th
Frontend, wireframe, UX, UI						
API Server, Authentication/Authorization,						
Broker, Router, Gateway						
Docker and Orchestration						

*th in month

Contribution

Open source projects with warm, welcoming communities keep people coming back for years. Many people form lifelong friendships through their participation in open source.

What does “open source” mean?

When a project is open source, that means anybody can view, use, modify, and distribute your project for any purpose. These permissions are enforced through an open source license.

Open source is powerful because it lowers the barriers to adoption, allowing ideas to spread quickly.

A common misconception about contributing to open source is that you need to contribute code. In fact, it’s often the other parts of a project that are most neglected or overlooked. You’ll do the project a huge favor by offering to pitch in with these types of contributions!

Even if you like to write code, other types of contributions are a great way to get involved with a project and meet other community members. Building those relationships will give you opportunities to work on other parts of the project.

Contact

Github: https://github.com/ParaffinIoT

Email: iokloud.com@gmail.com

Telegram ID: @pumpindex

Telegram Chat: https://t.me/joinchat/AuKmG05CNFTz0bsBny9igg