目录

第五部分 开源工具和方案

目前，MEC 的研究主要集中在标准化和接口标准化中，还没有统一标准化的 MEC 源码级别的解决方案。本章节参考边缘计算相关产业和学术机构成立的研究项目，从 MEC 仿真器相关工具、边缘计算开源平台和社区、代表性的商业边缘计算平台，以及商业边缘计算硬件产品等角度进行全面介绍。

1 相关平台工具

本章节将对 MEC 仿真实现或者搭建实验环境可能用到的主要工具和 SDR 和 IoT 平台进行介绍。

1.1 仿真平台

参考下表列出了用于仿真高级网络的工具和库，用于物联网，雾计算和边缘计算。

标题	简介	资源地址
iFogSim	雾计算，边缘计算和 IoT 的高性能开源工具包，用于建模和模拟边缘计算，物联网和雾计算的网络。	https://github.com/Cloudslab/iFogSim
CloudSim	云计算建模和仿真框架的基础架构和服务，其他扩展插件包括 Cloud2Sim、WorkflowSim、DynamicCloudSim、RealCloudSim、CloudReports、CloudAuction、FederatedCloudSim 等	https://github.com/Cloudslab/cloudsim
EmuFog	大型雾计算可扩展和可扩展仿真基础架构	https://github.com/emufog/emufog
FogTorch	物联网应用的 QoS 感知部署的工具	https://github.com/di-unipi-socc/FogTorch
cloonix	网络模拟平台，可以用图形化的方式构造网络拓扑，可以设置链路的延迟和丢包率	https://github.com/clownix/cloonix
imunes	集成的多协议网络模拟器/仿真器	https://github.com/imunes/imunes
contiki	专门针对物联网或者无线传感器网络应用的操作系统和协议栈，类似的系统 TinyOS	https://github.com/contiki-os/contiki
Cooja	cooja 是 contiki 网络模拟器。cooja 允许对 contiki motes 的网络进行模拟	https://github.com/contiki-ng/cooja
CupCarbon	智能城市和物联网无线传感器网络（SCI-WSN）模拟器	https://github.com/bounceur/CupCarbon http://www.cupcarbon.com/
DSA	物联网设备，服务和应用程序的开源物联网平台和“工具包”	http://iot-dsa.org/ https://github.com/IOT-DSA/docs/wiki
Mininet	由一些虚拟的终端节点（end-hosts）、交换机、路由器连接而成的一个网络仿真器，采用轻量级的虚拟化技术设计。	https://github.com/mininet
Netkit	基于 Linux 的开源网络模拟器，netkit-ng 是 netkit 的分支，为最新的 debian 内核和文件系统带来支持	http://netkit-ng.github.io/
Node-RED	一个基于浏览器、可拖放流和连接节点，并能将 IoT 设备与应用程序集成的工具	https://github.com/node-red
Zetta	基于 Node.js 的开源平台，用于创建跨地理分布的计算机和云运行的物联网服务器。Zetta 结合了 REST API，WebSocket 和响应式编程。	https://www.zettajs.org/ https://github.com/zettajs/zetta
Unetlab	以 ubuntu 为基础设计的一个超强网络实验模拟器，iou-web 的升级版。商用版本 EVE-NG（全称 Emulated Virtual Environment - NextGeneration）	http://www.routereflector.com/unetlab/ https://github.com/dainok/unetlab http://www.eve-ng.com/

Shadow	独特的离散事件网络模拟器，模拟运行真实的应用程序，比如 Tor，以及在一台机器上由数千个节点组成的分布式系统。	https://shadow.github.io/ https://github.com/shadow/shadow
GNS3	网络软件模拟器。 它允许组合的虚拟和实际设备，用于模拟复杂的网络。	https://gns3.com/ https://sourceforge.net/projects/gns-3/
NS3	一系列离散事件网络模拟器，包括 ns-1、ns-2 和 ns-3	https://www.nsnam.org https://github.com/nsnam
IoTivity	开源的软件框架，用于无缝的支持设备到设备的互联	https://iotivity.org/ https://github.com/iotivity/iotivity
Kaa	物联网平台，Kaa Enterprise 提供企业版物联网平台	https://www.kaaproject.org/ https://github.com/kaaproject/kaa
Cloud IoT Core	一种完全托管式服务，可以帮助用户安全地连接和管理大规模的 IoT 设备	https://cloud.google.com/iot-core/ https://cloud.google.com/iot/docs/quickstart
CORE	Common Open Research Emulator（CORE）是一种用于在一台或多台计算机上仿真网络的工具。	https://www.nrl.navy.mil/itd/ncs/products/core https://github.com/coreemu/core https://www.nrl.navy.mil/itd/ncs/projects
Meta-ACRN	ACRN 是 Linux 基金会发布的开源项目，这是一个专为物联网和嵌入式设备设计的管理程序	https://projectacrn.org https://github.com/intel/meta-acrn
OpenMQTT	OpenMQTTGateway（MQTT 网关）项目的目标是将一种不同技术集中在一个网关上，通过减少所需专有网关的数量并将不同技术的奇异性隐藏在一个简单而广泛的通信协议中来实现。	http://docs.openmqttgateway.com/#/ https://github.com/1technophile/OpenMQTTGateway
MQTT	机器对机器（M2M）“物联网”连接协议。它被设计为一种非常轻量级的发布/订阅消息传递。	http://mqtt.org/ https://github.com/mqtt
OpenNetworkLinux	开放网络 Linux 是 Linux 发行版，用于“裸机”交换机，即由商品组件构建的网络转发设备。开放网络 Linux 是开放计算项目的一部分。	http://www.opennetlinux.org/ https://github.com/opencomputeproject/OpenNetworkLinux https://www.opencompute.org/projects
FRRouting	FRRouting（FRR）是用于 Linux 和 Unix 平台的 IP 路由协议套件，其中包括 BGP，IS-IS，LDP，OSPF，PIM 和 RIP 的协议守护程序。	https://frrouting.org/#downloads https://github.com/FRRouting/frr
IO Visor	开发和共享虚拟化内核 IO 服务的功能，以实现跟踪，分析，监视，安全和网络功能	https://www.iovisor.org https://github.com/iovisor
OpenAMP	AMP 系统应用软件的开源软件框架	https://github.com/OpenAMP

iFogSim 仿真器

iFogSim 仿真器是一个采用不同资源管理和调度技术的评估环境，能够实现不同环境和条件下的跨边缘和云资源的应用程序调度策略。通过边缘设备、云数据中心和网络连接进行仿真，iFogSim 能够对用于边缘计算的资源管理策略进行评估，主要包括延时、能耗、网络瓶颈和操作成本等。iFogSim 支持的主要应用程序模型是 Sense-Process-Actuate 模型。在种仿真模型中，传感器将数据发布到 IoT 网络，由边缘云中的应用程序进行处理该传感器的数据，得到实时行为信息最后传递给执行器[1]。

iFogSim 与 CloudSim 协同工作，CloudSim 是一个广泛使用的库，用于模拟基于云的环境和资源管理。CloudSim 层使用 iFogSim 处理雾计算组件之间的事件。以下是模拟雾网络所需的 iFogSim 类：

- 雾设备
- 传感器

- 执行器
- 元组
- 应用
- 监控边缘
- 资源管理服务

iFogSim 的物理拓扑结果如下图 5-1-1 所示。iFogSim 物理实体主要包括[2]：

1）雾设备(FogDevice)，即边缘云设备：该类指定边缘云的硬件特征与其他边缘云、传感器和执行器之间的连接关系。此外，该类由 CloudSim 的 PowerDatacenter 类扩展而来，其属性包括可访问的内存、处理器、存储大小、上下通信能力等。

2）传感器(Sensor)：代表 IoT 传感器的实体，该类指定传感器特征和从其连通到输出的属性，以及连接到该传感器后边缘云的参考属性和它们连接的时延。

3）执行器(Actuator)：对网络连接特性进行建模，该类指定执行器所连接的网关和该连接的时延。

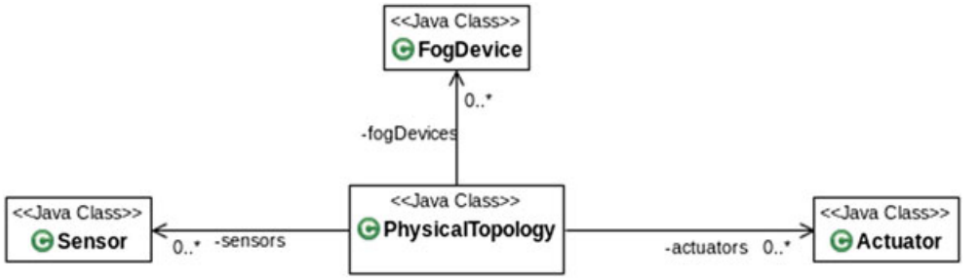

图 5-1-1 iFogSim 的物理拓扑结构图

1.2 SDR 平台

SDR LTE 开源平台如下表所示[3]：

平台	简介
OAI	OAI 是由欧洲的 Eurocom 组织开发并维护的开源 SDR LTE 平台，是目前最为完善的开源 SDR LTE 平台。 **1. 主要特征** - 开源项目 - 按照 3GPP 协议同时实现了 EPC，eNB 和 UE - 项目持续更新，目前已经支持 Release 10 - 支持支持 FDD/TDD，1.4, 3, 5, 10, 15 和 20 MHz 带宽 （目前 5 MHz 和 10 MHz 带宽运行比较稳定） - 项目包含仿真平台和 SDR 硬件实现平台。仿真平台包含链路级仿真平台，如 dlsim，系统级仿真平台，如 oaisim。 - 项目支持多种硬件设备，如 USRP，bladeRF 和 EXMIMO 等 - 目前 eNB 代码较稳定，UE 侧代码不太稳定 **2. 应用场景** - 各种链路级仿真，系统级仿真 - OAI eNB + OAI UE without S1 - OAI EPC + OAI eNB + OAI UE - Third-party EPC + OAI eNB + OAI UE - OAI EPC + OAI eNB + 商用终端（COST UE） - Third-party EPC + OAI eNB + 商用终端（COST UE）

	3. 相关链接 - 官方网站 http://www.openairinterface.org/ - 项目教程 https://twiki.eurecom.fr/twiki/bin/view/OpenAirInterface/WebHome - 代码获取 https://gitlab.eurecom.fr/oai/openairinterface5g
srsLTE	srsLTE 是由 SoftwareRadioSystems 实现的开源软件，起初只实现了 LTE 下行物理层链路功能 **1. 主要特征** - 开源项目 - 按照 3GPP 协议实现了 eNB 侧物理层下行的功能 - 目前只支持 Release 8 - 只支持 FDD SISO 模式, 1.4, 3, 5, 10, 15 和 20 MHz 带宽 - 项目支持 USRP, bladeRF 等硬件 - 产品稳定，代码优化较好，可以用作 SDR 开发的库 **2. 应用场景** - srs eNB + srs UE （物理层下行链路） **3. 相关链接** - 官方网站 http://www.softwareradiosystems.com/ - 代码获取 https://github.com/srsLTE/srsUE 和 http://github.com/srsLTE/srsLT
srsENB	SoftwareRadioSystems 公司开源了其全协议栈的 srsENB 平台，参考平台测试 srsENB+srsUE+Amarisoft MME[4] **1. 主要特征** - 开源项目 - 完全按照 3GPP 协议实现了全协议栈 UE(srsUE)和全协议栈的 eNB(srsENB) - 目前只支持 Release 8 - 只支持 FDD TM1 和 TM2 传输模式, 1.4, 3, 5, 10, 15 和 20 MHz 带宽 - 项目支持 USRP, bladeRF 等硬件 - 产品稳定，代码优化较好，可以用作 SDR 开发的库 **2. 应用场景** - Third-party MME+ srsENB + srs UE - Third-party MME+ Third-party ENB + srs UE - Third-party MME+ srsENB + Third-party UE - Third-party MME+ srsENB + 商用终端（手机、LTE 数据卡） **3. 相关链接** - 官方网站 http://www.softwareradiosystems.com/ - 代码获取 https://github.com/srsLTE/srsUE 和 http://github.com/srsLTE/srsLTE
OpenLTE	OpenLTE 由之前摩托罗拉的一位工程师发起的开源项目。 **1. 主要特征** - 项目开源 - 项目只包含 EPC 和 eNB，而且只包含 FDD 模式 - 只能传输信令，不能传输业务 **2. 应用场景** - OpenLTE EPC + OpenLTE eNB 等 **3. 相关链接** - 代码获取 1 http://openlte.sourceforge.net/ - 代码获取 2 https://sourceforge.net/projects/openlte/
Amarisoft	Amarisoft 是由 Fabrice Bellard 等实现的商业软件，是目前性能最好的 SDR LTE 平台。 **1. 主要特征** - 项目不开源 - 完全按照协议实现 3GPP LTE 协议的 eNB, EPC 和 UE - eNB 支持 LTE Release 13, UE 支持 LTE Release 12 - 支持 FDD/TDD, 1.4, 3, 5, 10, 15 和 20 MHz 带宽 - 项目留有硬件接口，对硬件没有要求（支持各种不同的硬件） - 软件优化较好，产品能在笔记本上运行

	- Amari UE 100 能在一台电脑上模拟并发送 500 个 UE 的数据 - 最近项目还在 eNB 侧添加了对 NB-IoT 协议的支持 **2. 应用场景** - 项目不开源，应用场景受软件授权的限制。 **3. 相关链接** - 官方网站 http://www.amarisoft.com/
Open-Source Long-Term Evolution (LTE) Deployment (OSLD)	**1. 主要特征** - 支持包括提供“用于构建基站和移动终端的模块化，提供开源 LTE 库 - ALOE（抽象层和开放操作环境），提供了用于开发开源 LTE 系统的 DSP 框架 - FlexNets 灵活的无线通信系统和网络 - Xenomai 提供 Linux 上具有硬实时功能，并且包括一些针对 MATLAB 和 GNU Octave 的通道模型 **2. 应用场景** - 为移动终端和基站开发模块化的 LTE 库，并提高 ALOE 的可访问性，以低成本构建复杂的无线电系统。 **3. 相关链接** - 官方网站 https://sites.google.com/site/osldproject/ - 方网站 https://nlnet.nl/project/osld/ - https://github.com/agelonch/aloe
GnuRadio	**1. 主要特征** - GNU Radio 是一个免费的软件开发工具包，提供信号处理模块以实现软件定义的无线电和信号处理系统。 - GNU Radio 模块：gr-lte、gr-lpwan、gr-dab、gr-fbmc、gr-gfdm、gr-radar、gr-specest、gr-cbmc、gr-drm、gr-classifier、gr_channelsounder。 - gr-lte 是一个 GNURadio 模块，gr-lte 专注于构建 LTE 接收器-特别是 LTE 信号的接收，同步和解码。 **2. 应用场景** - 以与外部 RF 硬件一起使用，以创建软件定义的无线电，如 GnuRadio + OpenLTE + SDR 搭建 4G LTE 基站。 - 在类似仿真的环境中无需硬件。 **3. 相关链接** -代码获取 1 https://github.com/kit-cel/ -代码获取 2 https://github.com/gbaier/gr_channelsounder -介绍 https://en.wikipedia.org/wiki/GNU_Radio 该项目是在德国卡尔斯鲁厄技术学院（KIT）的通信工程实验室（CEL）发起，网址为 http://www.cel.kit.edu。 - FlexNets http://flexnets.upc.edu；http://flexnets.upc.edu/trac/ - GNU Radio http://gnuradio.org - OSSIE from Wireless@VT (http://ossie.wireless.vt.edu/) - libLTE https://github.com/agelonch/libLTE - https://xenomai.org/
O-RAN（开放无线接入网）	O-RAN 开源社区将基于 O-RAN 联盟制定的相关规范和接口标准，C-RAN 联盟和 xRAN 论坛合并之后的产物。 **1. 主要特征** - 接口开放化 实现原有封闭接口的开放，降低区域性单一厂商的依赖性，鼓励创新，降低成本。 - 软件开源化 推动无线协议栈开源，共享代码，降低研发成本，让产业企业把更多精力聚焦在核心算法和差异化功能软件的研发上。 - 硬件白盒化 将传统 BBU 硬件用通用 COTS 服务器代替。而 RRU 部分，引入软件定义无线电技术和通用硬件，进行代替。 - 网络智能化 RAN 开放和解耦之后，可以引入大数据、人工智能等方法，实现复杂组网环境下的高效运维管理，提升频谱资源的利用率，降低网络能耗。

	2. 应用场景 - 推动 RAN 接口开放化、硬件白盒化、软件开源化、网络智能化，以此打破传统封闭的 RAN 构架，从而降低 RAN 部署成本，提升 RAN 敏捷性和加速创新。 **3. 相关链接** - 官方网站：https://www.o-ran.org/ - http://wiki.oran-osc.org/display/ORAN - https://gerrit.o-ran-sc.org/r/q/status:open
OMEC	**1. 主要特征** - OMEC 包括以下 EPC 和计费组件： SGW-C，PGW-C（包括嵌入式 PCEF），SGW-U，PGW-U，MME，HSS，HSS 数据库，路由功能，PCRF，转发策略控制 SDN 控制器，CTF，CDF，SGX 计费路由器 - CLI，日志记录和 VNF 的统计接口 API - 基于 DPDK 的流量生成器，用于测试 S1u 和 Sgi 用户平面 - 部署自动化，用于启动核心网络 VNF，关联的网络，程序包安装，设置，x 和配置 **2. 应用场景** - OMEC 是第一个功能齐全，可扩展的高性能开源 EPC。 **3. 相关链接** - 官方网站： https://www.opennetworking.org/omec/?utm_referrer=https://www.section.io/blog/telco-cooperation-accelerates-5g-edge-computing/ - 源码地址：https://github.com/omec-project

OpenAirInterface(OAI)，又称 OpenAirInterface5g，是欧洲 EURECOM 组织发起并维护的一个开源 SDR LTE 项目，其主要优势在于用软件实现了完整的 3GPP 协议，同时能够结合软件定义无线电组件(SDR)实现 4G LTE 基站。

OAI 的实现思想可简单概括为： PC 机通过软件实现物理层和介质访问层功能，同时，实现 3GPP 协议中的各汇聚和控制协议，然后将生成的 IP 数据通过 Linux IP 协议栈进行发送，而此时 eNode B 和 MME 之间通过各自 IP 进行连接和数据交换。目前，OAI 软件平台已经被作为无线电通信技术研究与实现的验证平台。

OpenAirInterfaceTM（OAI）源代码分为两个项目：

1) OAI Radio Access Network (OAI-RAN)

2) OAI Core Network (OAI-CN)

资源地址	描述
https://github.com/OPENAIRINTERFACE https://www.openairinterface.org/	官方网站
https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/GetSources https://gitlab.eurecom.fr/oai/openairinterface5g/wikis/OpenAirKernelMainSetup https://blog.csdn.net/wxsdr/article/details/53691763	官方安装指导
https://gitlab.eurecom.fr/oai/openairinterface5g/	OAI-RAN 项目实现了 4G LTE 和 5G 无线电接入网络。NodeB 和用户设备（UE）均已实现。
https://github.com/openairinterface	OAI-CN 项目实现 4G LTE 演进分组核心（EPC）和 5G 核心网。
https://github.com/OPENAIRINTERFACE/openair5g-cn	Openair5g-cn 是有关 3G 演进分组核心网的 3GPP 规范的实现。
https://github.com/OPENAIRINTERFACE/openair-cn	Openair-cn 是有关 3GPP 规范的实现 演进的分组核心网，这意味

	着它包含了网络元素：MME，HSS，S-GW ，P-GW
https://github.com/OPENAIRINTERFACE/openair-cn-cups	PGW-C 和 SPGW-U 的控制用户平面分离

OAI 按照 3GPP 协议完全实现了 LTE 的 EPC，eNB 和 UE，它的整体架构基本与 LTE 的协议栈。下图是 OAI 官方的一张结构图。可以看到左边是 UE 的结构图，包含 PHY，MAC，RLC，PDCP，RRC 和 NAS 等功能。但是这里强调下，由于 OAI 组织把开发的重点放在 EPC 和 eNB 上，UE 的代码是未经调试的，所以实际使用起来非常不稳定。但是经过我们的测试，OAI UE 的物理层下行还是计较稳定的。中间是 eNB 的架构图，右侧是 EPC。由于我主要接触的是 OAI eNB 和 OAI UE，对 EPC 接触也不多，所以我主要介绍 OAI 的 eNB 和 UE，即 RAN 部分。按照这个分类方法，OAI 也把项目分成了两部分，即 RAN 部分和 EPC 部分。

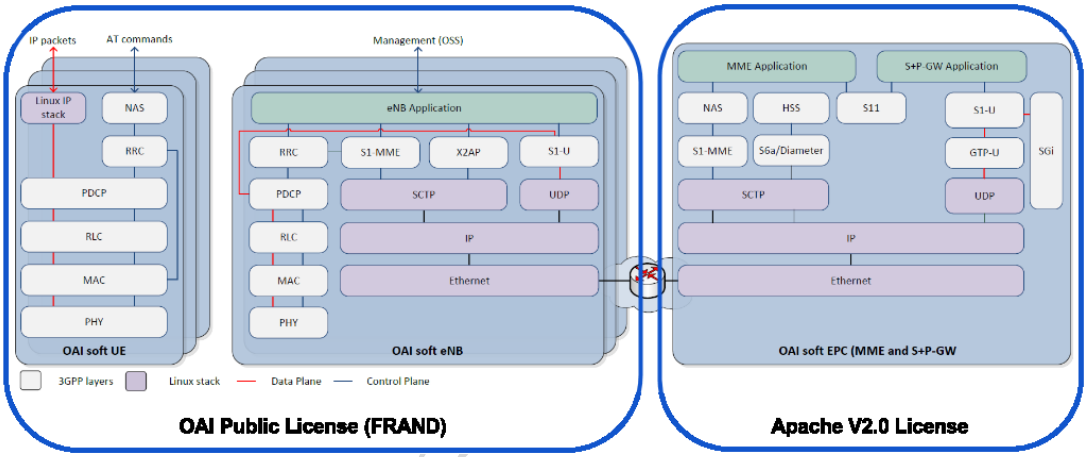

图 5-1-2 OAI 协议流

OpenAirInterface 主要包含四个部分：

OpenAir0：无线嵌入式系统设计，包含了一些硬件相关的设计文件和 firmware 之类。

OpenAir1：基带信号处理，包含了一些物理层的功能模块，例如 OFDM，调制解调，信道估计，编解码等等。

OpenAir2：中间层介入协议，包括在 PC 上通过 Linux 的 IP 网络设备驱动与 MPLS 的互联开发第二层协议栈。

OpenAir2：无线网络，包括为全 IP 蜂窝与 IP/MPLS 网状而开发的第三层协议栈。

1.3 IoT 平台

常见的物联网开源项目[5]，如 OpenIoT 、IoTSys、OpenRemote、openHAB、Device Hive、Kaa、The Thing System、Eclipse SmartHome、ThingBox、ThingSpeak、Zetta、AllJoyn、Eclipse Hono、Eclipse IoT 等、如下表所示：

平台	简介
OpenIoT	**1. 介绍** OpenIoT 是物联网的一个创新开源平台，包括了一些独特的功能，诸如基于云计算来组合各种重要的物联网服务。 **2. 主要特性** - 收集和处理世界各个角落传感器的数据，包括物理设备、传感器处理算法、社交媒体处理算法等等； - 将各个传感器的数据流导入云计算架构中； - 动态发现/查询传感器以及它们的数据；

	- 组合并传递基于大量传感器数据的物联网服务； - 物联网数据的可视化展示（表格、图形等）； - 优化 OpenIoT 中间件和云计算架构的资源。 **3. 相关链接** - 官网地址：https://openiot.in/ - 资源地址：https://github.com/OpenIotOrg/openiot； https://github.com/OpenIotOrg/openiot/wiki
TencentOS tiny	**1. 介绍** TencentOS tiny 是腾讯面向物联网领域开发的实时操作系统，提供精简的 RTOS 内核，内核组件可裁剪可配置，可快速移植到多种主流 MCU (如 STM32 全系列)及模组芯片上。 **2. 主要特性** - 小体积：最小内核 RAM 0.6KB，ROM 1.8KB 典型 LoraWAN 及传感器应用：RAM 3.3KB，ROM 12KB - 低功耗：休眠最低功耗低至 2 uA 支持外设功耗管理框架 - 丰富的 IoT 组件：集成主流 IoT 协议栈 多种通信模组 SAL 层适配框架； - 支持 OTA 升级 提供简单易用端云 API，加速用户业务接入腾讯云 - 可靠的安全框架：多样化的安全分级方案 均衡安全需求&成本控制 - 良好的可移植性：内核及 IoT 组件高度解耦，提供标准适配层 提供自动化移植工具，提升开发效率 - 便捷的调试手段：提供云化的最后一屏调试功能 故障现场信息自动上传云平台，方便开发人员调试分析 **3. 相关链接** - 源码地址 1：https://github.com/Tencent/TencentOS-tiny
Kaa IoT Platform	**1. 介绍** Kaa 物联网开发平台由美国迈阿密的团队开发，是功能丰富的开放和高效的物联网云平台。 **2. 主要特性** - 事件系统，发现设备的高级功能 - 日志收集 - 配置和分组 - 发送提醒 - 数据分布式管理 - Transport abstraction - 支持个商业条目和多个应用配置 **3. 相关链接** - 官方网址：https://www.kaaproject.org - 源码地址：https://github.com/kaaproject/kaa；https://github.com/kaaproject
HarmonyOS	**1. 介绍** 华为鸿蒙系统（HarmonyOS）是基于微内核的全场景分布式 OS，可按需扩展，实现更广泛的系统安全，主要用于物联网，特点是低时延，甚至可到毫秒级乃至亚毫秒级，由华为技术有限公司开发 **2. 主要特性** - 分布式架构首次用于终端 OS，实现跨终端无缝协同体验； - 确定时延引擎和高性能 IPC 技术实现系统天生流畅； - 基于微内核架构重塑终端设备可信安全； - 通过统一 IDE 支撑一次开发，多端部署，实现跨终端生态共享。 **3. 相关链接** - 资源地址： https://github.com/Awesome-HarmonyOS/HarmonyOS
SiteWhere	**1. 介绍** 它是提供设备数据的摄取，存储，处理和集成的另一个开源 IoT 平台。SiteWhere 运行在 Apache Tomcat 提供的核心服务器上。它提供高度调整的 MongoDB 和 HBase 实现。 **2. 主要特性** - 在单个 SiteWhere 实例上运行任意数量的 IoT 应用程序 - Spring 提供了核心配置框架 - 用 MQTT，AMQP，Stomp 和其他协议连接设备 - 通过自注册，REST 服务或批量添加设备

	- 与第三方集成框架（如 Mule AnyPoint）集成 - 默认的数据库存储是 MongoDB - Eclipse Californium 进行 CoAP 消息传递 - InfluxDB 用于事件数据存储 - Grafana 可视化 SiteWhere 数据 - HBase 用于非关系数据存储 **3. 相关链接** - 资源地址：https://github.com/sitewhere - 资源地址：https://github.com/sitewhere/sitewhere
EMQX	**1. 介绍** EMQ X 代理是面向 IoT，M2M 和移动应用程序的完全开源，高度可扩展，高可用性的分布式 MQTT 消息传递代理，可以处理数千万个并发客户端。 EMQ X Broker：开源物联网 MQTT 消息中间件 EMQ X Enterprise：企业级物联网 MQTT 消息平台 EMQ X Platform：电信级多协议物联网接入平台 EMQ X Edge：开源轻量边缘计算消息中间件 **2. 主要特性** - 海量物联网设备一站式连接，3G/4G/5G&NB-IoT 全网络支持，MQTT&CoAP 多协议支持，TLS/DTLS 多重网络安全，X.509 证书等多种身份认证。 - 高并发低时延，大规模分布式。千万级并发连接，百万级消息吞吐，毫秒级消息时延。大规模分布式，高可用集群架构，弹性伸缩部署，5G 时代大型物联网应用首选技术方案。 - 强大规则引擎，快速应用集成。强大的内置规则引擎，一站式数据提取、过滤与转换。灵活集成 SQL、NoSQL、时序数据库，与 Kafka 流处理中间件。快速应用集成与持续创新。 - 边缘到云端，云端到跨云部署。从资源受限的边缘计算设备，到私有云、混合云和公共云之上，到跨域、跨 IDC 与跨多云，EMQ X 支持物理机、VM、容器/K8S 跨平台任意部署。 **3. 相关链接** - 官网地址：https://www.emqx.io/ - 资源地址：https://github.com/emqx/emqx
Apache Edgent	**1. 介绍** Apache Edgent 是一种编程模型和具有微内核风格的运行时，可嵌入到网关和小型的物联网设备中。Apache Edgent 能用于对来自器材、车辆、系统、应用、设备和传感器（例如树莓派或智能手机）的连续数据流进行实时分析。通过与集中式分析系统协同工作，Apache Edgent 可在整个物联网生态系统中提供高效、以及从中心到边缘的实时数据分析。 **2. 主要特性** - 物联网（IoT）：分析分布式物联网设备和移动设备上的数据，以便降低传输数据的成本和在设备上提供本地反馈； - 嵌入在应用程序服务器实例中：实时分析应用程序服务器错误日志，而不会影响网络通信量 - 服务器机房和机房：实时分析机器运行状况，而不会影响网络流量或带宽有限 **3. 相关链接** - 官网地址：https://edgent.apache.org/ - 资源地址：https://edgent.incubator.apache.org/docs/home.html https://github.com/apache/incubator-edgent https://edgent.apache.org/docs/downloads

公司级 IoT

公司	IoT	官网
Akamai	IoT Edge Connect	https://www.akamai.com/us/en/products/performance/iot-edge-connect.jsp
Google	Cloud IoT Edge	https://cloud.google.com/solutions/iot/?hl=zh-cn
亚马逊	AWS Greengrass	https://aws.amazon.com/greengrass/

微软	Azure IoT Edge	https://azure.microsoft.com/en-us/services/iot-edge/ https://github.com/Azure/iotedge
阿里云	Link IoT Edge	https://help.aliyun.com/document_detail/102727.html?spm=a2c4g.11186623.6.554.f9196091DwJQB4 https://github.com/aliyun/linkedge-thing-access-sdk-c

1.4 参考文献

[1] Gupta H, Vahid Dastjerdi A, Ghosh SK, Buyya R. iFogSim: A toolkit for modeling and simulation of resource management techniques in the Internet of Things, Edge and Fog computing environments. *Softw Pract Exper*. 2017;47:1275–1296. https://doi.org/10.1002/spe.2509.

[2] Redowan Mahmud and Rajkumar Buyya, Modelling and Simulation of Fog and Edge Computing Environments using iFogSim Toolkit, Fog and Edge Computing: Principles and Paradigms, Wiley Press, New York, USA, 2018 (in press).

[3] 软件无线电 SDR LTE 平台简介[EB/OL]. 2016-11-03. https://blog.csdn.net/jxwxg/article/details/53026659

[4] srsENB+srsUE+Amarisoft MME 测试[EB/OL]. 2017-06-22. https://blog.csdn.net/jxwxg/article/details/73603577

[5] 21 个物联网开源项目[EB/OL]. https://www.linux.com/NEWS/21-OPEN-SOURCE-PROJECTS-IOT/

2 开源平台

边缘计算系统平台的基本目的是提供一种降低延迟、提供数据处理实时性等针对特定边缘计算应用的一种架构和软件栈。由于边缘计算应用场景的复杂性，以及终端设备和用户的多样性，导致目前边缘计算系统平台的研究明显趋向碎片化。

本章节对当前新兴的边缘计算开源平台，如 LF Edge、StarlingX、KubeEdge、OTE-Stack、CORD、Cloudlet、ParaDrop、Mbed 等开源平台进行简单介绍。

2.1 LF Edge

LF Edge 由 Linux 基金会成立，旨在帮助建立一个统一的边缘计算(edge networking)市场，并为未来的边缘计算项目提供一个通用框架。项目主要包括五个开源项目，它们将作为 LF Edge 工作的核心，当前又添加了 Baetyl 和 Fledge 两个项目到 LF Edge 中。

1) EdgeX Foundry：源于戴尔/EMC，EdgeX Foundry 是一种与平台无关的框架，允许即插即用的集成微服务，可以在任何工业边缘网关上运行，也可以在传感器和云之间建立桥接层。

2) Home Edge Project：由 Samsung 提供，Home Edge Project 主要目标是为消费者物联网(consumer IoT)做些什么，它是一个适用于基于家庭的物联网设备的 Run-Anywhere 服务层。

3) Akraino Edge Stack：开源 Akraino Edge Stack 主要基于 AT&T 的代码贡献，旨在将类似的概念标准化，以用于除 IoT 网关之外的边缘电信和网络系统。采用云端的 Akraino Edge Stack，旨在实现资源调配自动化，为尝试使用后端云服务运行边缘计算服务的企业提供灵活性和可扩展性支持。

4) Project EVE：EVE 项目(Edge Virtualization Engine)。该项目由创业公司 ZEDEDA 提供，Project EVE 是一个边缘虚拟化引擎。其目的是让用户虚拟化边缘硬件，从而更有效地分配任务负载，减少延迟，实现应用程序开发与基础架构解决方案的解耦。

5) Open Glossary of Edge Computing：这是一个边缘计算术语参考文件，每一条技术术语都是理解关于边缘计算的讨论所必需的。它由一个合作小组维护，其成员包括 ARM、边缘计算提供商 Packet、爱立信、微服务提供商 CDN Rafray Systems 和边缘集中主机代管提供商 Vapor.io。目前最新版本为 v2.0.0。

6) BAETYL：OpenEdge 新版本，更名为百度智能边缘计算框架 BAETYL，聚焦在物联网边缘计算的云原生基础设施，具有平台中立、系统中立、网络中立的特点。

7) Fledge：Dianomic 的 FogLAMP 的新版本。专为工业边缘而设计，是工业运营商，系统集成商和设备提供商嵌入，部署。

2.1.1 Akraino

Akraino Edge Stack 是一个面向高性能边缘云服务的开源项目，并为边缘基础设施提供整体的解决方案。开源软件堆栈提供了关键基础架构，以实现高性能，减少延迟，提高可用性，降低运营开销，提供可伸缩性，满足安全需求并改善故障管理。提供关键的基础能力包括：

- 支持线性处理
- 支持高吞吐量
- 减少延迟
- 提高可用性
- 降低运营开销
- 可扩展性
- 解决安全性需求
- 改善故障管理

Akraino Edge Stack 社区专注于 Edge API，中间件，软件开发套件（SDK），并将允许与第三方云的跨平台互操作性。边缘堆栈还将支持边缘应用程序的开发，并创建带有虚拟网络功能（VNF）生态系统的应用程序。

目前最新 Akraino Release 1 版本，提供了针对边缘计算系统和应用程序进行了优化的高可用性云堆栈，并且“旨在改善企业边缘，OTT 边缘和运营商边缘网络的边缘云基础架构的状态”。Akraino R1 包含 11 个以上的蓝图系列，其中 19 个以上的特定蓝图正在开发中（如 R2 版本），以支持各种边缘用例，如下表所示。

Blueprint Family	Blueprint Species Name	Submitter	Release Target
Network Cloud	SDN Enabled Broadband Access (SEBA)	AT&T	R1
	Serverless	AT&T	R1
	Unicycle Blueprint (SR-IOV)	AT&T	R1
	Rover Blueprint	AT&T	R2
	Real Time Edge Media Processing	Radisys	R1
	Network Cloud and TF Integration	Juniper	R1
	OVS-DPDK Unicycle (Dell)	Ericsson	R1
Integrated Edge Cloud	IEC Type 1: small deployment	Arm	R1
	IEC Type 2: medium deployment	Arm	R1
Edge Light & IoT	ELIoT 2: LW Edge	Huawei	R1
	SD-WAN	Huawei	R1
Kubernetes Native Infrastructure for Edge	Provider Access Edge	Red Hat	R1
	Industrial Edge	Red Hat	R1
Micro MEC	Micro MEC Type 1	Nokia	R2
	Micro MEC Type 2	Nokia	R2
	Micro MEC Type 3	Nokia	R2
Radio Edge Cloud	Radio Edge Cloud	Nokia	R1
Far Edge Cloud	Starling X Far Edge Distributed Cloud	WindRiver	R1
TBD	Time Critical Edge Compute	Intel	R1

R1 版本提供了 10 个“现成的”系列，包括以下内容：

（1）无线电边缘云（REC，Radio Edge Cloud）：作为电信设备蓝图系列的一部分，REC 是电信级的容器边缘云平台。它主要设计为支持实时 RAN 智能控制器，能够使用外部应用程序控制 5G 无线电网络的各个方面。

（2）集成边缘云（IEC，Integrated Edge Cloud）类型 1 小型边缘（Small Edge）和 2 中型边缘（Medium Edge）：边缘云平台可以在网络边缘上实现新的功能和业务模型。以中小型边缘云部署的电信应用为目标，并支持 ARM 处理器和体系结构。

（3）网络云（Network Cloud）：支持从远程区域控制器进行硬件配置和多个边缘站点的全自动部署。R1 中的特定网络云系列中包括支持 SR-IOV 的 Unicycle；支持 OVS-DPDK 的 Unicycle；以及支持多种电信和企业边缘用例（包括 5G）的 Rover。

（4）StarlingX 远程边缘分布式云（StarlingX Far Edge Distributed Cloud）：基于英特尔和风河支持的 StarlingX 项目，该系列解决了购物中心，机场和体育馆等高密度位置的边缘和边缘用例，以支持这些事件和位置的增值服务（如缓存，处理和数据分析）。

（5）边缘轻量级和物联网（ELIOT，Edge Lightweight and IOT）：轻量级的 ELIOT 系列支持 IoT 网关和 uCPE（SD-WAN）的用例，包括工业物联网，智慧城市和 uCPE。ELIOT 实现了一个轻量级软件堆栈，可以部署在硬件容量有限的边缘节点上。

（6）Kubernetes 本地基础架构（KNI，Kubernetes-Native Infrastructure）：利用 Kubernetes 提供的最佳实践来大规模管理边缘计算堆栈，从基础设施到工作负载，在裸机或公共云上提供一致、统一的用户体验。

Akraino 项目涉及的范围从基础设施延伸至边缘计算应用，其范围可以划分为 3 个层面，如下图 5-2-1 所示。

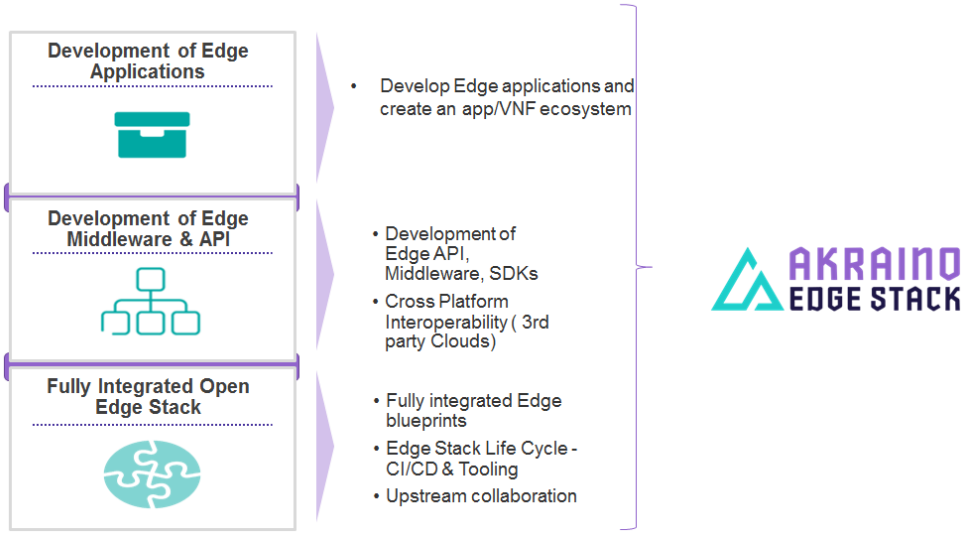

图 5-2-1 Akraino Edge Stack 软件堆栈范围

上层是应用部署，主要负责部署边缘应用并创建 APP/VNF 的边缘生态系统。

中间层是边缘中间件和 API，创建标准的边缘平台和中间件，并统一 API 和 SDK 接口。以接入现有互补性的开源边缘计算项目，如面向物联网的互操作性框架 EdgeX Foundry。

底层是 IaaS 基础设施层面中，提供一套开源软件栈用于优化基础设施，对接开源的边缘堆栈。如 OpenStack、Kubernetes 等。

Akraino 项目集成了众多的项目和软件，包括 Airship、Calico、Camunda、CI/CD、Ceph、CNI、Gerrit 、Jenkins、JIRA、Kubernetes、Nexus 3、ONAP、OpenStack、OVS-DPDK、SR-IOV、Tempest 等，参考如下图 5-2-2 Akraino 软件架构图。从总体结构上看类似 OPNFV 的集成项目，既包含顶层边缘应用程序，也包含中间与底层基础架构框架交互中的中间件和边缘 API，同时还包含管理底层基础架构的 Edge Stack 以及相应的生命周期管理、CI/CD 和工具集。

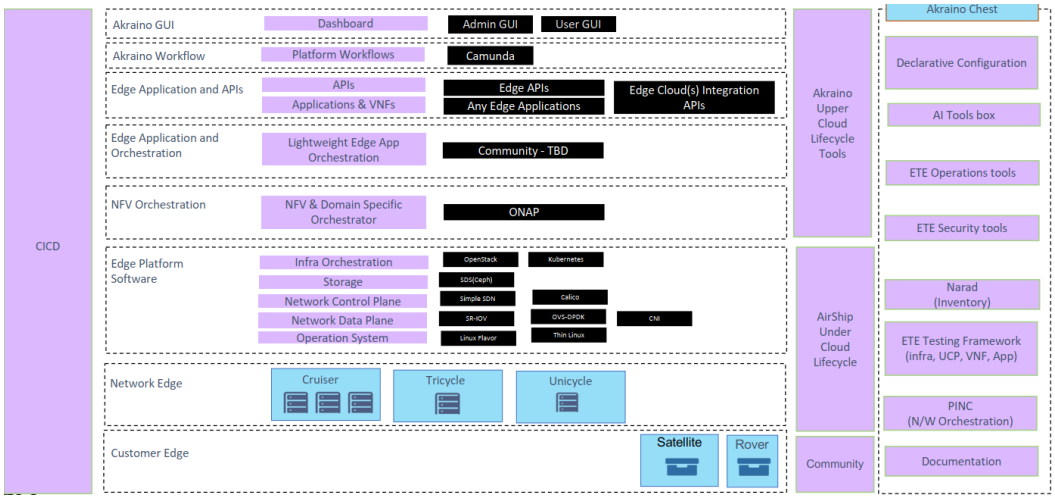

图 5-2-2 Akraino 软件架构图

上图中，在底层边缘网络（Network Edge）层分为 3 级 Pod，在此层中交付点（Pod，Point of Delivery）是将服务部署到边缘站点的方法。同时也边缘位置可以具有单个服务器或多个服务器（如 Customer Edge 层的 Rover 和 satellite），包

含在一个或多个机架中（如 Open19 标准化服务器架构）。由于 Akraino 使用声明性配置，因此交付点允许被边缘设备统一管理。Pod 可以使用 cookie-cutter 方法以更低的成本在更大规模（例如，10,000 多个位置）上部署。作为示例，下图 5-2-3 显示了各种 Pod，例如 cruiser-large Pod，tricycle-medium Pod，unicycle 单机架 Pod，satellite--多台服务器和 Rover 单台服务器。每个 Pod 都有不同的硬件组件，以满足特定的电信和企业用例。

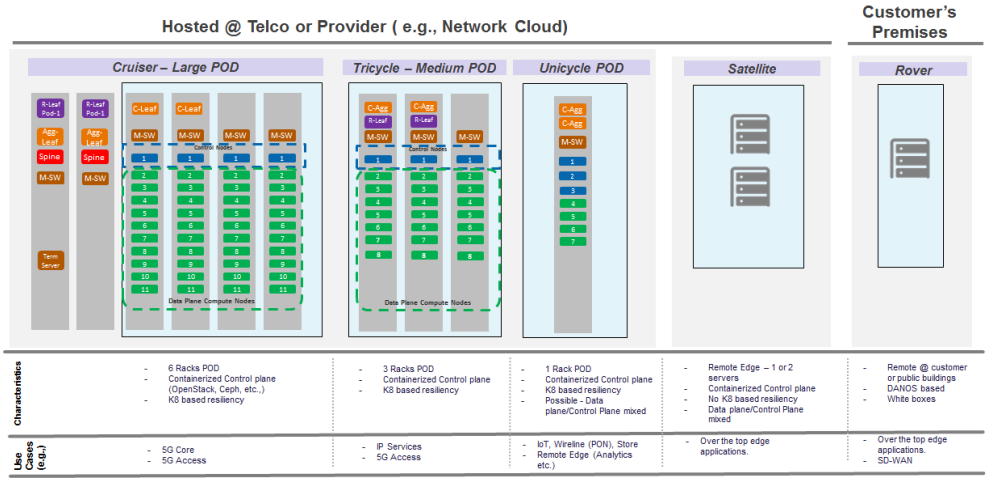

图 5-2-3 Akraino Pod 部署点

2.1.2 EdgeX Foundry

EdgeX Foundry 创造一个互操作性的、即插即用组件的物联网边缘计算的生态系统。通用的边缘微服务层，目标是通过支持用户在边缘计算节点上部署混合的即插即用微服务，提高互操作性，例如支持分析、数据编排、数据库、安全性、系统管理和服务。理想情况下，该项目需要基于 EVE 或 Akraino 运行，但是用户也可以使用 Linux 和 Docker 运行时。

EdgeX Foundry 的主要特性包括：

1) 提供灵活的微服务构架；

2) 服务具有高扩展性，能够根据设备能力和现实环境进行自由扩展，包括在多个边缘硬件处理器之间进行动态服务分配；

3) 以通用 API 的形式支持对移动设备的接口协议转化；

4) 在保证高扩展性的同时，具备工业级安全性、稳定性和可靠性。

EdgeX Foundry 的定位是通用工业 IOT 边缘计算通用框架。EdgeX Foundry 是一个开源微服务集合，由 4 个服务层和 2 个系统服务组成，其架构如图 5-2-4。自下而上分别为设备服务层、核心服务层、支持服务层和传输服务层。另外两个系统服务分别为安全服务和系统管理服务。

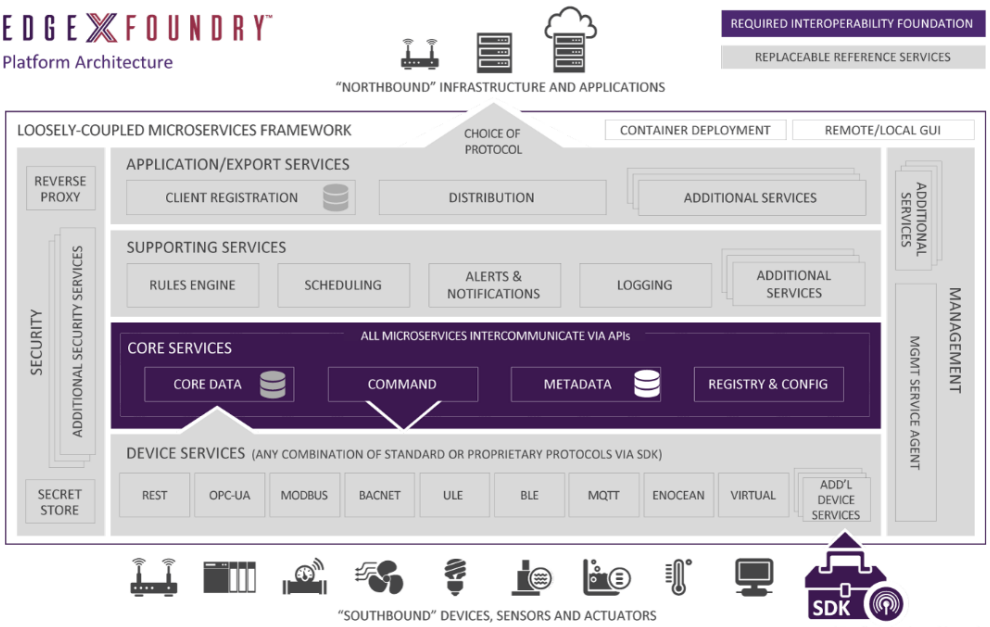

图 5-2-4　EdgeX Foundry 架构图

EdgeX Foundry 在构架中定义了“南侧”和“北侧”能力，其中，南侧包括所有的物联网物理设备以及与这些设备、传感器、执行器或者其他对象直接通信的边缘器件；而北侧则是负责数据汇总、存储、聚合、分析和转换的云平台以及负责与云平台通信的网络部分。EdgeX Foundry 架构图从下向上依次为设备服务层、核心服务层、支持服务层、开放服务层；这个顺序也是物理域到信息域的数据处理顺序。

核心服务层（Core Services Layer - CS）

核心服务层介于北向与南向之间，这里所谓的北向即是上文所述信息域，南向即是上文所述物理域。核心服务层非常简单，但却是 EdgeX Foundry 框架内非常重要的一环。核心服务层主要由以下组件组成：

- 注册和配置（Configuration and Registration）：为其他 EdgeX Foundry 微服务提供关于 EdgeX Foundry 内相关服务的信息，包括微服务配置属性。
- 核心数据（Core Data）：持久性存储库和从南侧对象收集的数据的相关管理服务。
- 元数据（Metadata）：提供配置新设备并将它们与其拥有的设备服务配对的功能。
- 命令（Command）：处理北向应用发往南向设备的请求；当然该服务还会处理框架内其他微服务发往南向设备的请求，如本地的分析服务。

支持服务层（Supporting Services Layer - SS）

支持服务层包含多个微服务：

- 边缘分析和边缘智能
- 提供给 EdgeX Foundry 本身的服务
- 通用软件服务，如日志、调度、以及数据清理（Scrubbing）
- 规则引擎

- 告警和消息通知微服务
- 本地分析能力（目前的实现也叫简单规则引擎）

传输服务层（Export Services Layer - ES）

EdgeX Foundry 的一个设计原则是独立于其他系统运行，即原则 3：EdgeX Foundry 必须支持存储转发能力。因为部署在网关上的边缘计算服务经常会处于离线环境，需要能独立运作监控管理某些不需要外部监控的设备和传感器。因此，EdgeX Foundry 必须能够不接入北向系统独立运行相当长一段时间。传输服务层的任务就是将边缘侧采集的数据传输到云端。

ES 层的微服务提供一下服务：

- 提供注册服务接口给网关外部客户端，注册接受从南向对象采集的数据。
- 提供数据将在何时发向何处的信息
- 为其传输的数据提供数据的格式

ES 服务层目前提供以下微服务：

- 客户端注册
- 分发
- AWS IoT Core
- Azure IoT Core
- Google IoT Core

设备服务层（Device Services Layer - DS）

设备服务层负责与南向设备交互。设备服务是与南向设备或物联网对象交互的边缘连接器，包括但不限于：报警系统，家庭和办公楼中的暖气和空调系统，灯光，任何行业的机器，灌溉系统，无人驾驶飞机，目前自动化的运输，如一些铁路系统，自动化工厂，家用电器。未来，还可能包括无人驾驶汽车和卡车，交通信号灯，全自动快餐设施，全自动自助式杂货店，以及从病人身上读取健康数据的设备。

设备服务不仅可以连接一个设备，也可以连接多个设备。DS 所管理的设备可以是简单设备，也可以是设备网关，设备管理器，设备聚合器等。DS 层通过设备原生的通信协议与设备进行通信，然后将数据转成 EdgeX Foundry 内部通信的标准数据结构，再发送给核心服务层和其他服务层。EdgeX Foundry 提供一个设备服务软件开发工具（SDK），通过命令行的方式生成设备服务的基本框架。

目前，EdgeX Foundry 的设备服务层包含以下微服务：

- 虚拟设备服务
- BACnet 设备服务
- Modbus 设备服务
- SNMP 设备服务
- 低功耗蓝牙（BLE）设备服务
- 串口设备服务

系统管理和安全服务

系统管理服务提供安装、升级、启动、停止和监测控制 EdgeX Foundry 微服务的功能。安全服务用以保障来自设备的数据和对设备的操作安全

2.1.3 EVE

边缘虚拟化引擎（EVE）是一种安全的设计操作系统，支持在现场部署的计算设备上运行边缘容器。这些设备可以是 IoT 网关，工业 PC 或通用加固型计算机。统称为“边缘节点”。EVE 利用 Type-1 虚拟机管理程序，它不直接支持类似 POSIX 的传统应用程序和流程，可以部署在裸机设备硬件上。在 EVE 上运行的所有应用程序都必须表示为边缘容器，它可以是虚拟机，容器或 Unikernel。除轻量级虚拟化引擎外，Project EVE 还提供了零信任安全框架。

基于 EVE 的边缘设备具有以下功能：

- 设备资源的高效利用
- “默认安全”配置部署文件
- 可托管部署在任何操作系统的虚拟机中
- 在虚拟机和容器中托管许多应用
- 通过 Unikernel 的无服务器功能
- 可扩展的集中式管理，适用于远距离的许多设备，并托管许多应用程序
- 整个软件堆栈的支持远程更新
- 远程控制所有设备资源，包括 CPU，内存，网络和设备端口
- 自动修补安全更新
- 可自动连接到一个或多个公共云
- 内置的 mesh 网络功能可实现边缘到边缘的数据流
- 内置云网络支持使用公共云中可用的 VPN 标准技术

EVE 包含四个主要组件，如图 5-2-5：

- Type-1 虚拟机管理程序(hypervisor)
- 操作系统服务(Operating system services)
- 设备连接和管理界面(Device connectivity and management interfaces)
- 边缘容器运行时(Edge container runtime)

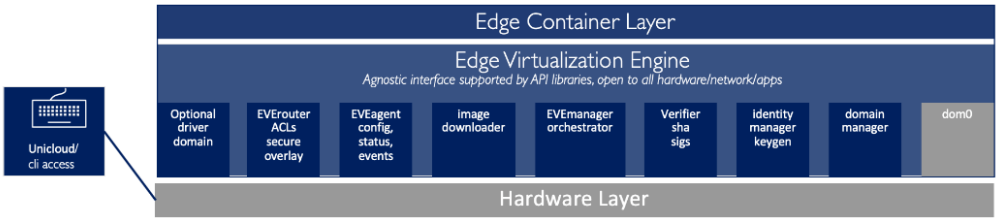

图 5-2-5 EVE 主要组件

- **虚拟机管理程序（Hypervisor）**

EVE 包括基于 Xen 的 Type-1 虚拟机管理程序，从而使其可以在任何受支持的硬件平台上运行。

- **系统服务（System services）**

这些服务使 EVE 及其托管的应用程序，能够为所有设备资源和实例进行自我更新，身份管理，安全性和网络服务。

- **设备连接和管理界面（Device connectivity and management interfaces）**

包括设备网络接口，用于以太网和无线网络的驱动程序，以及用于将 EVE 连接到集中式管理服务的 API。

- **边缘容器运行时（Edge container runtime）**

包括域管理，实例流程，虚拟化 IO，实例间网络和远程实例控制台。

EVE 体系架构包含一些功能，这些功能可管理整个设备，EVE 本身以及对运行多个应用程序实例的支持。如下图 5-2-6 所示 Project EVE 架构。

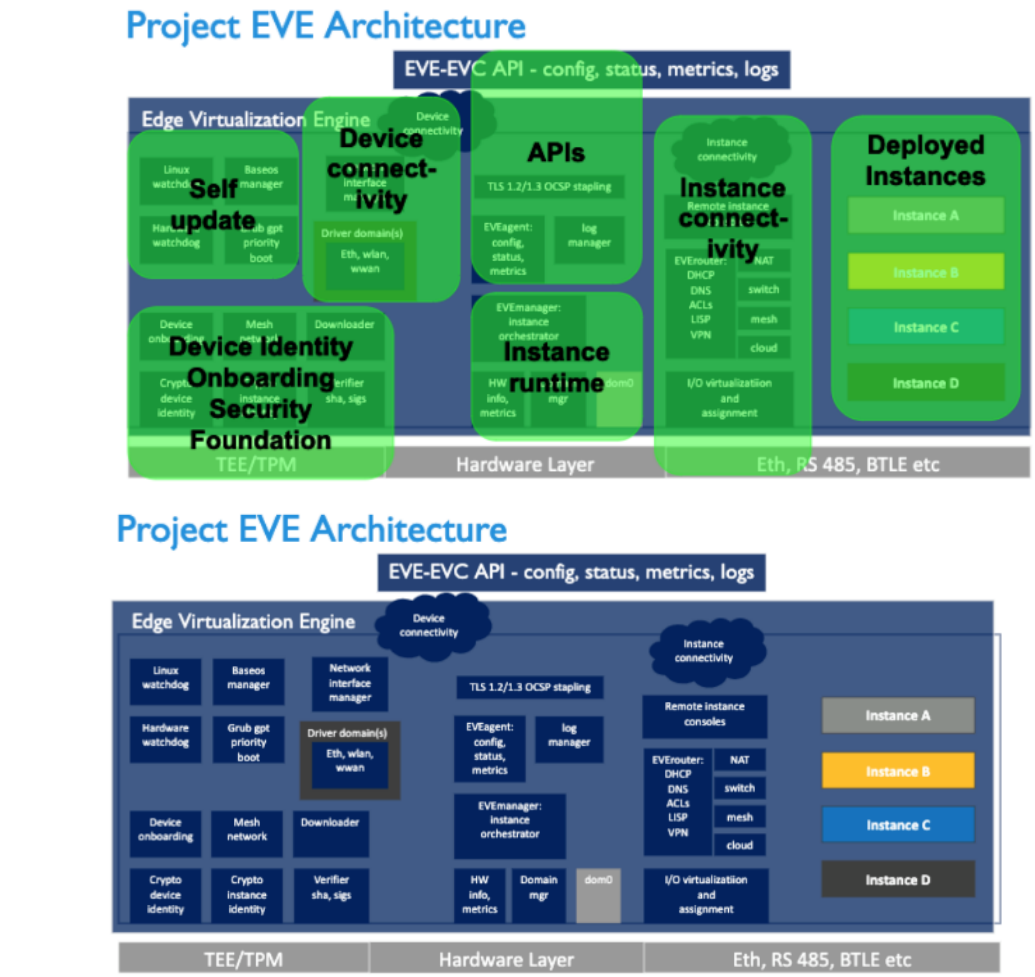

图 5-2-6 Project EVE 架构

上图组件和功能：

- 设备身份，接入，安全基础（Device Identity, Onboarding, Security Foundation）
- 自主更新（Self Update）

- 设备连接（Device Connectivity）
- EVE API 管理（EVE Management APIs）
- 实例运行时（Instance Runtime）
- 实例连接（Instance Connectivity）
- 实例部署（Deployed Instances）

2.1.4 Home Edge

Home Edge Project 专注于驱动并实现一个健壮，可靠，智能的家庭边缘计算开源框架、平台和生态系统，这些框架可在日常生活中的各种设备上运行。为了成功地加速边缘计算服务生态系统的部署，“家庭边缘项目”将为用户提供一个可互操作，灵活且可扩展的边缘计算服务平台，该平台具有一组 API，这些 API 也可以与库运行时一起运行。

该项目的平台体系架构以下模块组成，如下图 5-2-7。

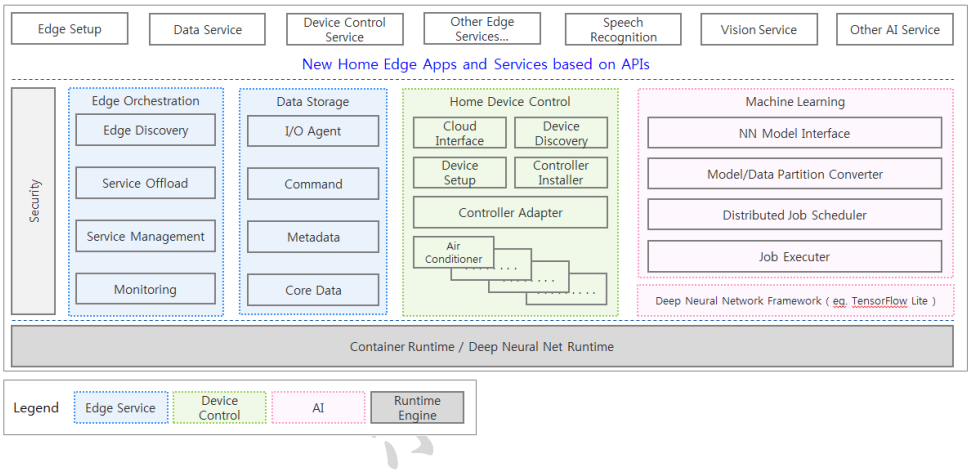

图 5-2-7 Home Edge 架构

边缘编排模块（Edge Orchestration Module）

- 边缘发现：主动发现用户家庭网络中的家庭边缘设备。
- 服务迁移：将服务重新迁移到其他家庭边缘设备，以实现设备和/或服务的负载均衡。
- 边缘配置：在家庭边缘设备上配置网络信息和用户配置文件。
- 服务管理：管理服务的生命周期和副本。
- 监控：检查并通知用户边缘网络中的家庭边缘设备及其连接的家庭边缘设备和服务状态。

数据存储模块（Data Storage Module）

- 核心数据：在 Home Edge 设备上从其连接的家庭边缘设备（消费电子产品，传感器和事物）和服务中收集的那些数据，并提供持久性存储。
- 元数据：所连接的家庭边缘设备（消费类电子产品，传感器和事物）和服务的标识/配置文件/数据。
- I/O 代理：提供用于访问数据存储的 API。
- 命令：提供一种通过基于规则的方法，从家庭边缘设备的数据存储中获取特定数据的方法。

家用设备控制模块（Home Device Control Module）

- 云接口：提供云服务与用户家庭网络中每个家庭边缘设备之间的接口。
- 设备发现：从家庭边缘设备中查找已连接的家庭设备，首次发现连接的家庭设备时，其客户端服务由控制器安装。
- 设备配置：配置连接的家庭设备所需的网络信息，以使其加入用户的家庭网络。
- 控制器安装器：安装发现的已连接家用设备的客户端服务。
- 控制器适配器：提供统一的 API，用于控制连接的家用设备。
- 家用设备客户端：控制连接的家用设备（例如，用户的冰箱，洗衣机，灯泡，门锁，HVAC 和温度传感器）。

机器学习模块（Machine Learning Module）

- 神经网络模型接口：为使用（分布式）神经网络处理推理和识别提供 API。
- 模型分区转换器：用于分布式神经网络处理的模型划分。
- 分布式任务调度程序：调度和分配分布式任务。
- 任务执行器：使用深度神经网络框架在家庭边缘设备上执行分布式任务。

安全模块（Security Module）

- 安全模块提供了家庭边缘设备和服务提供的安全功能，如安全登录，证书，AAA，消息协议的加密/解密等。

深度神经网络框架（Deep Neural Network Framework）

- 深度神经网络框架为机器学习提供了数据流编程框架，如 TensorFlow Lite 等。

2.1.5 Baetyl

Baetyl 旨在将云计算能力拓展至用户现场，提供临时离线、低延时的计算服务，包括设备接入、消息路由、消息远程同步、函数计算、设备信息上报、配置下发等功能。百度智能边缘计算框架 BAETYL 同时具备两大 AI 能力，一是 AI as a Function，BAETYL 的 Python runtime 可以支持 scikit-learn 类型的数据分析模型；二是视觉 AI 能力，支持边缘侧视频接入、视频抽帧、图像标注、数据上行、知识下行（模型下发）、AI 推断等，这些能力将在此次项目捐赠后全面开源。值得一提的是，BAETYL 和百度智能边缘 BIE（Baidu IntelliEdge）云端管理套件配合使用可以达到云端配置、边缘运行的效果，满足各种边缘计算场景的需求。

Baetyl 优势：

- 屏蔽计算框架：Baetyl 提供主流运行时支持的同时，提供各类运行时转换服务，基于任意语言编写、基于任意框架训练的函数或模型，都可以在 Baetyl 中执行。
- 简化应用生产：智能边缘 BIE 云端管理套件配合 Baetyl，联合百度云，一起为 Baetyl 提供强大的应用生产环境，通过 CFC、Infinite、EasyEdge、TSDB、IoT Visualization 等产品，可以在云端轻松生产各类函数、AI 模型，及将数据写入百度云天工云端 TSDB 及物可视进行展示。

- 服务按需部署：Baetyl 推行容器化和模块化，各模块独立运行互相隔离，开发者完全可以根据自己的需求选择部署。
- 支持多种平台：Baetyl 支持多种软硬件平台，比如 X86 和 ARM 架构的 CPU，Linux 和 Darwin 操作系统等。

Baetyl 致力于为以下方面建立开放源代码框架：

- 从物联网设备到分布式集群甚至嵌入式设备，将不同形式的硬件抽象到统一的容器环境中；
- 支持开放应用程序模型，包括 OCI 容器和无服务器模式，例如 FaaS 和流技术；
- 提供与 k8s 原语兼容的标准化远程管理模型。

Baetyl 架构，如图 5-2-8，目前 Baetyl 开源了如下几个官方模块：

- baetyl-agent：提供 BIE 云代理服务，进行状态上报和应用下发。
- baetyl-hub：提供基于 MQTT 的消息路由服务。
- baetyl-remote-mqtt：提供 Hub 和远程 MQTT 服务进行消息同步的服务。
- baetyl-function-manager：提供函数计算服务，进行函数实例管理和消息触发的函数调用。
- baetyl-function-python27：提供加载基于 Python2.7 版本的函数脚本的 GRPC 微服务，可以托管给 baetyl-function-manager 成为函数实例提供方。
- baetyl-function-python36：提供加载基于 Python3.6 版本的函数脚本的 GRPC 微服务，可以托管给 baetyl-function-manager 成为函数实例提供方。
- baetyl-function-node85：提供加载基于 Node8.5 版本的函数脚本的 GRPC 微服务，可以托管给 baetyl-function-manager 成为函数实例提供方。

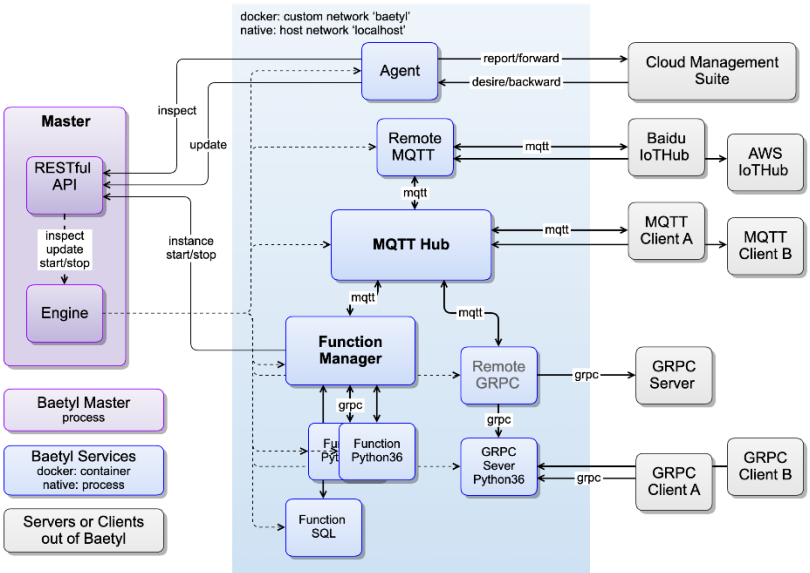

图 5-2-8 Baetyl 架构

2.1.6 Fledge

Fledge 是面向工业领域的开源框架和社区，重点关注关键操作，预测性维护，态势感知和安全性。由 Dianomic 贡献，以前称为“ FogLAMP”。 Fledge 架构集成了 IIoT，传感器和现代机器，以及工业“brown field”系统，分布式控制系统（DCS、Distributed Control Systems）、程序逻辑控制器（PLC，Program Logic Controllers）、监控和数据采集（SCADA，Supervisory Control and Data Acquisition）。所有这些组件共享管理和应用程序 API。通过使用一致的 RESTful API 来开发，管理和保护 IIoT 应用程序，Fledge 创建了一个统一的解决方案。

Fledge 与 EVE 项目和 Akraino 紧密合作。Project EVE 为 Fledge 应用程序和服务提供系统和编排服务以及容器运行时。Fledge 的垂直行业（制造，能源等）开始部署 5G 和专用 LTE 网络。通过使用 Akraino 蓝图，可以对 Fledge 应用程序和服务进行充分管理，因为它们利用了 5G 和专用 LTE 网络。

FogLAMP 是一种用于物联网（IoT）的开放式传感器到云数据架构，它提供了可扩展的，安全的，强大的基础框架，用于从传感器收集数据，在边缘处理数据以及将数据传输到 Historian 和其他管理系统。FogLAMP 被实现为微服务的集合，其中包括：

- 核心服务，包括安全性，监视和存储
- 数据转换和告警服务
- 南向服务：从传感器和其他 FogLAMP 系统收集数据
- 北向服务：向 Historian 和其他系统传输数据
- 边缘数据处理应用

下图 5-2-9 显示了 FogLAMP 的体系架构。

- 蓝色组件是插件

  插件是轻量级的模块，可扩展 FogLAMP。有多种类型的插件：南向，北向，存储引擎，过滤器，事件规则和事件传递机制。插件可以用 python（用于快速开发）或 C++（用于高性能）编写。

- 绿色组件是微服务

  它们可以共存于相同的操作环境中，或者可以分布在多个环境中。

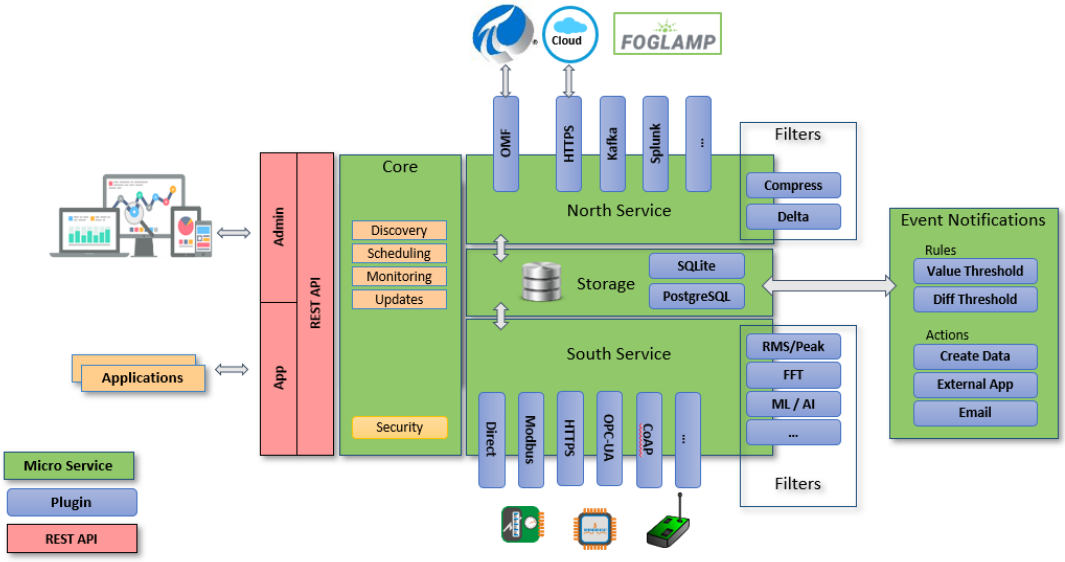

图 5-2-9 FogLAMP 架构

- FogLAMP 核心

核心微服务协调所有的 FogLAMP 操作。任何时候都只能激活一项核心服务。核心功能包括：

1) 调度：灵活的调度以启动程序。

2) 配置管理：维护所有 FogLAMP 组件的配置。维护所有 FogLAMP 组件软件更新。

3) 监视：监视所有 FogLAMP 组件，如果发现问题（例如微服务无响应），尝试自愈。

4) REST API：为所有组件功能的公开外部管理和数据 API。

5) 备份：FogLAMP 系统备份和还原功能。

6) 审计日志：维护系统日志以便进行日志审计。

7) 证书存储：维护不同组件的安全证书，包括南向服务，北向服务和 API 安全。

8) 用户管理：维护 FogLAMP 管理员的身份验证和权限信息。

9) 资源查询：对已存储资源数据的查询。

- 存储层

存储微服务提供两个主要功能：

1）维护 FogLAMP 配置和运行时状态

2）存储/缓冲资源数据。存储引擎的类型是可插入的，因此在占用空间较小的安装中，可以选择 SQLite 插件，或者在并发请求数量较多、且占用空间较大的 Postgresql 安装。在微型安装中，例如在边缘设备上，内存临时存储可能是最佳选择。

- 南向微服务

南向微服务在边缘设备（例如传感器，执行器或 PLC 和 FogLAMP）之间提供数据和元数据的双向通信。较小的系统可能在边缘设备上安装了此服务。南向组件通常部署为始终运行的服务，该服务不断等待新数据。或者，可以将它们部署为单个任务，这些任务会定期启动，收集数据和启动。

- 北向微服务

北向微服务在 FogLAMP 平台与本地或云中的大型系统之间提供数据和元数据的双向通信。大型系统可能是私有和公共云数据服务，专有解决方案或具有较大占用空间的 FogLAMP 实例。通常将北向组件部署为一次性任务，这些任务会定期向上 spin up 并发送已批处理的数据，然后向下 spin down。它们也可以部署持续运行的服务。

- 过滤器

筛选器是用于修改流经 FogLAMP 数据流的插件。它们可以部署在入口（在南向服务中）或出口（在北向服务中）。通常，入口过滤器用于转换或过滤数据，出口过滤器用于减少流向北向管道和基础设施的流量，即通过压缩或减少流出的数据。可以在“管道”中应用多个过滤器，配置后，管道可以应用于多个南向或北向服务。

- 事件引擎

事件引擎维护零个或多个规则/操作。每个规则订阅所需的资源数据，并对其进行评估。如果规则触发，则执行其关联的操作。

- REST API

FogLAMP API 提供了管理 FogLAMP 以及与其中的数据进行交互的方法。

- 图形用户界面

GUI 允许管理 FogLAMP。所有 GUI 功能都通过 REST API 进行，因此 FogLAMP 也可以通过脚本或其他管理工具进行管理。GUI 包含以下页面：

健康状况：查看服务是否响应。查看流入和流出 FogLAMP 的数据

资源读取：FogLAMP 中的数据分析

南向：管理南向服务

北向：管理北向服务

通知：管理事件引擎规则和传递机制

证书存储：管理证书

备份与还原：备份/还原 FogLAMP

日志：查看系统，通知，审核日志记录信息

2.1.7 资源列表

LF Edge 资源列表如下：

标题	简介	资源地址

LF Edge	为边缘计算建立一个开放的，可互操作的框架，以独立于硬件，芯片，云或操作系统。	https://github.com/lf-edge https://www.lfedge.org/ https://wiki.lfedge.org/
Glossary	边缘计算开放词汇表、边缘计算领域相关的术语	https://www.lfedge.org/projects/openglossary/ https://github.com/lf-edge/glossary
Akraino Edge Stack	开源软件堆栈，该堆栈支持针对边缘计算系统和应用程序优化的高可用性云服务	https://www.lfedge.org/projects/akraino/ https://wiki.akraino.org/ https://wiki.akraino.org/display/AK/Documentation https://gerrit.akraino.org/r/q/status:open https://github.com/akraino-edge-stack
EdgeX Foundry	为 IoT 边缘计算构建通用的开放框架，IoT Edge 生态系统开放式互操作平台。	https://wiki.edgexfoundry.org/ https://github.com/edgexfoundry https://www.edgexfoundry.org/ https://wiki.edgexfoundry.org/ https://hub.docker.com/u/edgexfoundry
Project EVE	开源边缘虚拟化引擎（EVE，Edge Virtualization Engine）	https://github.com/lf-edge/eve https://www.lfedge.org/projects/eve/ https://wiki.lfedge.org/display/EVE/Project+EVE
Home Edge	致力于推动并实现一个强大、可靠、智能的家庭边缘计算框架、平台和生态系统	https://www.lfedge.org/projects/homeedge/ https://wiki.lfedge.org/display/HOME/Home+Edge+Project
Baetyl	百度提供的“OpenEdge”，它将云计算，数据和服务无缝地扩展到边缘设备。	https://www.lfedge.org/projects/baetyl/ https://github.com/baetyl/baetyl https://wiki.lfedge.org/display/LE/Baetyl https://baetyl.io/zh/
Fledge	面向工业领域的开源框架和社区，重点关注关键操作，预测性维护，态势感知和安全性	https://www.lfedge.org/projects/fledge/ https://wiki.lfedge.org/display/LE/Fledge
FogLAMP	工业物联网（IIoT）开源项目，并且是基本的雾计算组件。FogLAMP 使用可插拔的模块化架构来轻松连接任何/所有传感器，机器和 IIoT 设备，管理其数据并将其转发给历史学家（例如 OSIsoft 的 PI），企业系统和云。	http://foglamp.readthedocs.io/ https://foglamp.readthedocs.io/en/master/ https://github.com/foglamp/FogLAMP https://foglamp.readthedocs.io/en/1.3/01_introduction.html https://foglamp.readthedocs.io/en/latest/foglamp_architecture.html https://dianomic.com/ https://www.osisoft.com/
Edge Computing Landscape	边缘计算系统生态圈	https://www.stateoftheedge.com/ https://www.stateoftheedge.com/projects/landscape/ https://github.com/State-of-the-Edge/landscape
CNCF	云原生计算基金会（CNCF、Cloud Native Computing Foundation）是一个开源软件基金会，它致力于云原生（Cloud Native）技术的普及和可持续发展	https://landscape.cncf.io/ https://www.cncf.io/

2.2 StarlingX

StarlingX 是 Intel 和 WindRiver 开源的边缘计算项目。该项目是基于 WindRiver 的产品 Titanimu Cloud R5 版本基础上修改而来。Titanium Cloud 是基于 Openstack 专门针对 NFV 场景开发的产品。该产品具有 WindRiver 在实时操作系统多年的积累，自主开发的基于 DPDK 的 AVS（虚拟交换机），能够支撑电信云的高带宽，低时延的要求。

StarlingX 版本由五个基础设施服务组成，如图:

- 配置管理(Configuration Management)

配置管理功能在边缘云基础设施架构中变得非常重要，特别是在管理大量的远端节点的时候，因为有些远处的节点，不太方便直接对其进行配置。因此借助于配置管理功能点，可以方便地对远端的物理服务器进行配置管理，配置管理中包含了 CPU、GPU、内存、Huge pages，crypto/compression PCIE 配置等。

- 故障管理(Fault Management)

这个组件是可以统计报警和查看 log，并且同时包括了中心云和边缘云的物理资源和虚拟资源，并且在 Horizon 上都可以进行查看，监控的方面比 OpenStack 更广。

- 主机管理(Host Management)

这个组件可以检查虚拟主机的状态，并在主机关机的情况下尝试自动重启，并根据集群状态、关键进程、资源的阈值、物理主机的故障等来使用不同的调度策略来进行对虚拟机的重启。

- 服务管理(Service Management)

该功能点提供了服务的高可用，使用了多路通道来避免通信的断开和服务的脑裂问题，基于 StarlingX 本身服务的 active/passive 状态的切换来保障服务的高可用，并对服务的状态进行监控。

- 软件管理(Software Management)

从 kernel 到 OpenStack 服务的全栈软件包升级，该功能可以实现滚动升级，比如在需要对物理服务器关机的情况下实现对虚拟机的热迁移的情况，该功能在 StarlingX 中仅需要在 horizon 界面上进行操作，该热迁移可以自动把需要更新软件包主机上的虚拟机或者容器事先迁移到可用的主机，并在更新完成之后，再自动将资源分配到更新完成的主机上，该功能提供了对升级时候的虚拟机关机问题的生命周期管理的机制。

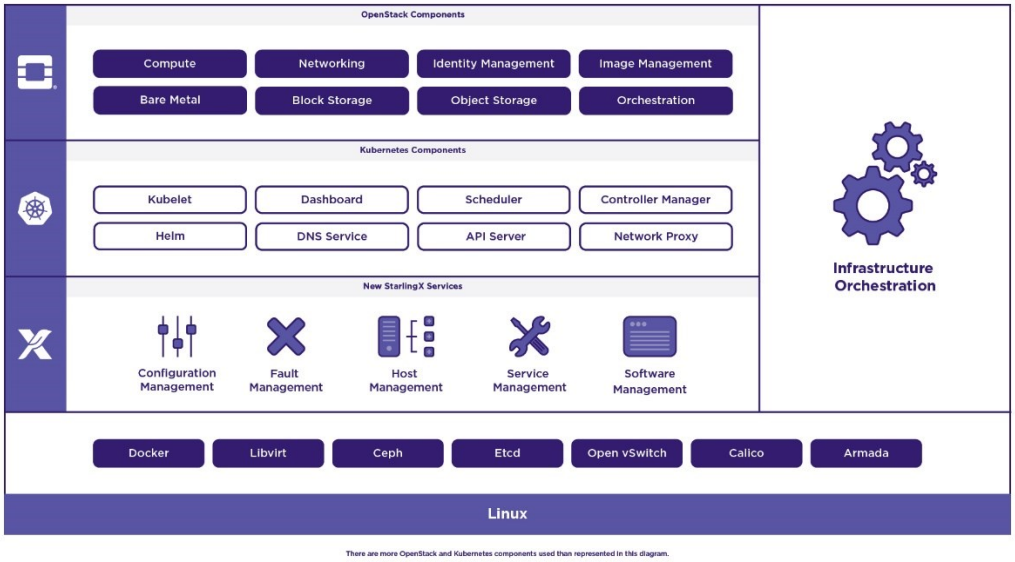

图 5-2-10 StarlingX 的整体架构

资源地址	标题
https://docs.starlingx.io/ https://wiki.openstack.org/wiki/StarlingX https://github.com/starlingx-staging https://opendev.org/starlingx https://www.starlingx.io/	StarlingX

2.3 KubeEdge

KubeEdge 即 Kube+Edge，顾名思义就是依托 K8s 的容器编排和调度能力，实现云边协同、计算下沉、海量设备的平滑接入。KubeEdge 是一个支持边缘计算开放平台的开源系统，用于将容器化应用程序编排功能扩展到 Edge 的主机。基于 kubernetes 构建，并为网络应用程序提供基础架构支持。云和边缘之间的部署和元数据同步。KubeEdge 的目标是创建一个开放平台，使能边缘计算，将容器化应用编排功能扩展到边缘的节点和设备，后者基于 kubernetes 构建，并为云和边缘之间的网络，应用部署和元数据同步提供基础架构支持。

KubeEdge 重点要解决的问题是云边协同、资源异构、大规模、轻量化、以及一致的设备管理和接入体验。KubeEdge 架构上分为三个部分，分别是云、边、端三侧。云端负责云上应用和配置的校验、下发，边缘侧则负责运行边缘应用和管理接入设备，设备端运行各种边缘设备。KubeEdge 完整的打通了边缘计算中云、边、设备协同的场景，整体架构如下图 5-2-11 所示。

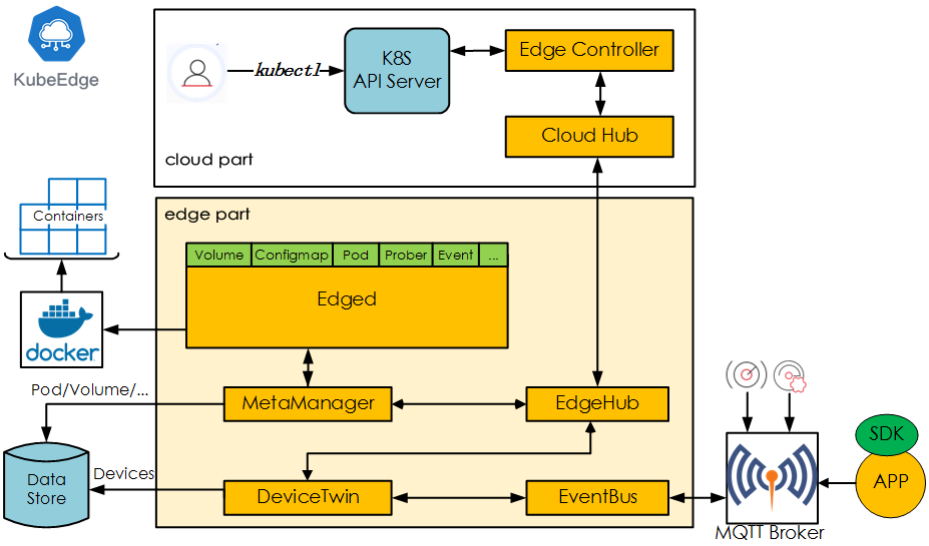

图 5-2-11 KubeEdge 架构图

KubeEdge 的边缘进程包含以下 5 个组件：

- Edged：管理 Edge 的容器化应用程序，重新开发的轻量化 Kubelet，实现 Pod，Volume，Node 等 Kubernetes 资源对象的生命周期管理。
- EdgeHub：Edge 的通信接口模块，多路复用的消息通道，提供可靠和高效的云边信息同步。它是一个 Web 套接字客户端，负责与 Cloud Service 进行边缘计算交互。
- EventBus：使用 MQTT 处理内部边缘通信，订阅来自于 MQTT Broker 的设备数据。它是与 MQTT 服务器（mosquitto）交互的 MQTT 客户端，为其他组件提供发布和订阅功能。
- DeviceTwin：它是处理设备元数据的设备的软件镜像，用于抽象物理设备并在云端生成一个设备状态的映射。此模块有助于处理设备状态并将其同步到云。它还为应用程序提供查询接口，因为它与轻量级数据库（SQLite）接口。
- MetaManager：它管理边缘节点的元数据，负责本地元数据的持久化，是边缘节点自治能力的关键。这是 edged 和 EdgeHub 之间的消息处理器。它还负责向轻量级数据库（SQLite）存储/检索元数据。

KubeEdge 的云端进程包含以下 2 个组件：

- CloudHub：云端的通信接口模块，部署在云端，接收 EdgeHub 同步到云端的信息。
- EdgeController：管理 Edge 节点。部署在云端，用于控制 Kubernetes API Server 与边缘的节点、应用和配置的状态同步。

资源地址	标题
https://github.com/kubeedge/kubeedge https://kubeedge.io https://kubeedge.readthedocs.io/en/latest/ https://docs.kubeedge.io/en/latest/	KubeEdge

2.4 OTE-Stack

OTE-Stack 是面向 5G 和 AI 的开源边缘计算平台，致力于实现多种边缘资源的统一接入 ，通过边缘资源的虚拟化，集群管理和智能调度，屏蔽底层异构的特性，提高资源利用率，降低接入门槛；作为边缘基础设施，支撑 Device-Edge-Center 算力的统一调度，为 AI 提供低延时，高可靠，成本最优的边缘算力支持。

OTE-Stack 核心特性：

- 大规模集群分层管理

通过标准接口，可以快速构建层次集群。从理论上讲，集群的数量是无限的，可以有效解决 5G 时代大规模移动边缘集群的管理和调度问题。

- 支持第三方集群扩展

原生支持 kubernetes 和 k3s。可以通过 cluster-shim 接入任意第三方集群。因此在 5G 时代，它可以与不同运营商的 MEC 平台的不同实现兼容。

- 轻量级集群控制器

仅需一个组件即可完成边缘集群快速接入。因此，它非常轻巧且易于使用。

- 边缘集群轻量自治

边缘集群只保存核心状态，确保轻量，即服从中心调度，又可实现自治。

- 自动容灾

由于集群分层设计，每个级别的集群控制器将自动获取备用节点。一旦与父节点的连接丢失，与中心的连接将通过备用节点恢复。

- 全局统一调度

通过集群控制器，可以将所有集群集成到统一调度中，并且可以实现边缘资源的全局最优利用。

- 多种虚拟机支持

OTE-Stack 利用 virtlet 处理基于 VM 的工作负载，并通过 CustomResourceDefinition 添加 VM 操作（启动，停止，安装等）。因此它支持可以统一编排的 KVM，Kata 容器和 Dockers。

- Kubernetes 原生支持

借助 OTE-Stack，用户可以像云中的传统 kubernetes 集群一样在边缘集群上编排 docker / VM。

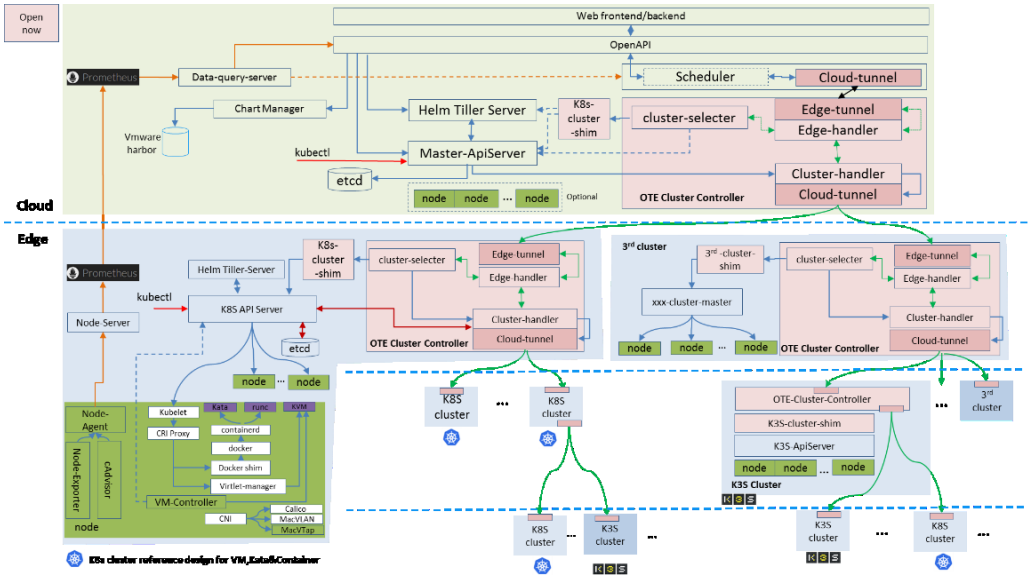

图 5-2-12 OTE-Stack 架构图

框架组件包括：WebFrontend、WebBackend、OpenAPI、Scheduler、ChartManager、EdgeTunnel、EdgeHandler、ClusterSelecter、ClusterHandler、CloudTunnel、k8s-cluster-shim、k3s-cluster-shim、NodeAgent、NodesServer、DataQueryServer、VMController。

资源地址	标题
https://ote.baidu.com/ https://github.com/baidu/ote-stack	OTE-Stack

2.5 CORD

CORD 是为网络运营商推出的开源项目，旨在利用软件定义网络（SDN）、网络功能虚拟化（NFV）和云计算技术重构现有的网络边缘基础设施，并将其打造成可灵活地提供计算和网络服务的数据中心。项目的目标是提供一个网络运营商的服务交付平台的参考实现。其核心输出包括一个软件平台、系列硬件规范和服务模型等。

CORD 的软件架构如图 5-2-13 所示，云平台管理项目 OpenStack 用以管理计算和存储资源，创建和配置虚拟机以及提供基础设施即服务（IaaS）功能。开源网络操作系统（ONOS）为网络提供控制平面，用于管理网络组件如白盒交换网络结构等，并提供通信服务。容器引擎 Docker 使用容器技术来实例化提供给用户的服务。服务控制平台 XOS 用于整合上述软件，以组装、控制和组合服务。

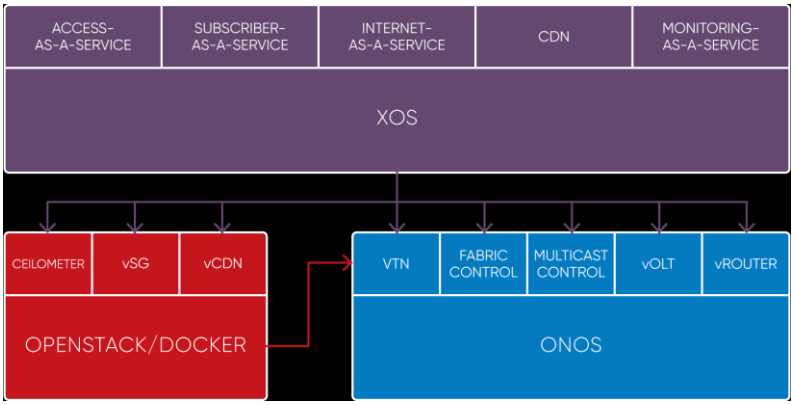

图 5-2-13 CORD 的软件架构

根据运营商网络组成和应用场景，CORD 分为三大部分：M-CORD、R-CORD、E-CORD 和 A-CORD。

1) 家庭宽带，Residential-CORD(简称 R-CORD)；

2) 企业宽带，Enterprise-CORD(简称 E-CORD)；

3) 移动宽带，Mobility-CORD(简称 M-CORD)；

4) 分析运营平台，Analytical-CORD(简称 A-CORD)；

服务	全称	场景	说明
vRouter	Virtual Router	CORD	虚拟路由器服务，当前版本的 vRouter 支持 BGP 协议。VRouter 服务可被配置的参数包括设备、端口、接口和 IP。 CORD 中的 vRouter 服务旨在使 CORD POD 能够与上游路由器进行路由和通信。
VTN	Virtual Tenant Network	CORD	CORD 上的虚拟租户网络服务（Overlay）。
vTR	Virtual Truckroll Service	R-CORD	用真实的用户和设备来测试网络连通性成本太高。这个服务就是用于模拟用户和设备测试连通性的。
vOLT	Virtual OLT	R-CORD	替代传统厂商的 OLT 设备的控制平面。提供用户认证，vlan 等功能。
vSG	Virtual Subscriber Gateway	R-CORD	为用户提供家庭网关功能。
VTN	Virtual Tenant Network	R-CORD	虚拟租户网络服务
vNaaS	virtual Network as a Service	E-CORD	虚拟网络即服务。位于 E-CORD global 节点
vEE	Virtual Ethernet Edge	E-CORD	虚拟以太网边缘设备。位于 E-CORD local 节点
vEG	Virtual Enterprise Gateway	E-CORD	虚拟企业网关。位于 E-CORD local 节点
vCPE	Virtual Customer Premise Equipment	E-CORD	功能上和 R-CORD 中 vOLT 类似。位于 E-CORD local 节点
PWaaS	PseudoWire as a Service	E-CORD	将购买该服务的企业直连到城域网。位于 E-CORD local 节点
epc-service	Evolved Packet Core-service	M-CORD	移动通信（4G、5G）核心网
vENB	Virtual eNodeB servcice	M-CORD	虚拟化的移动通信（4G、5G）基站
vSPGWC	Virtual Serving PDN Gateway-Control Plane Service	M-CORD	虚拟化的 PDN 网关的控制平面
vSPGWC	Virtual Serving PDN Gateway-User plane Service	M-CORD	虚拟化的 PDN 网关的用户平面

COMAC 平台中，如下图 5-2-14 所示。

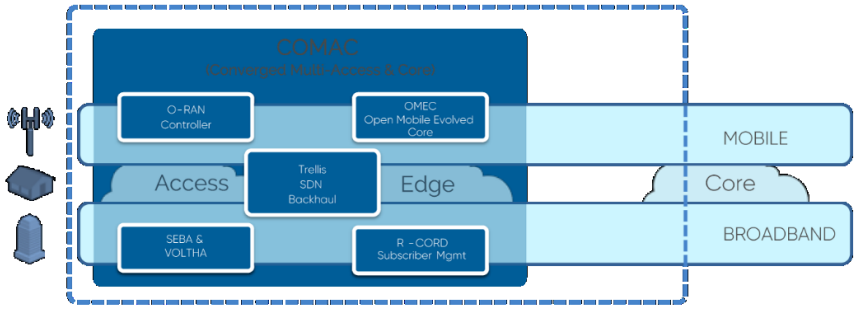

图 5-2-14 COMAC 平台

CORD 硬件架构，如下图 5-2-15 所示。

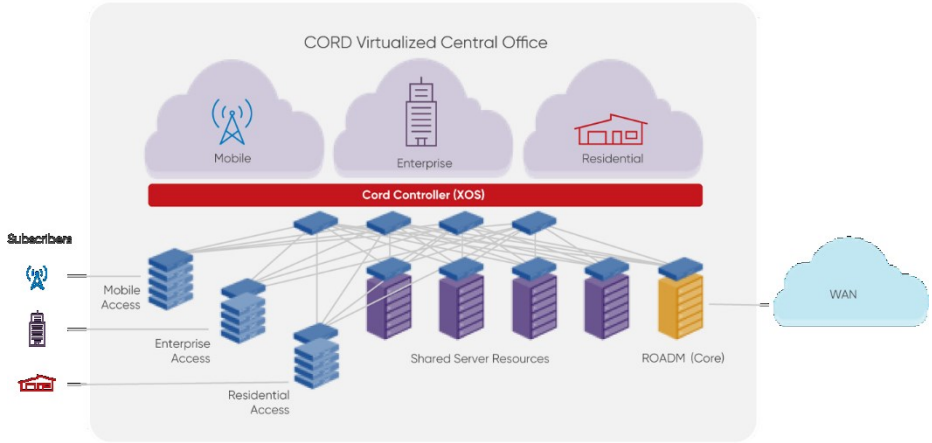

图 5-2-15 COMA 硬件架构

CORD 开源项目 OpenCORD（R）提供了许多可部署的、特定于 NFV 平台的用例，这些平台集成到基于 OPNFV 参考平台中的许多核心平台组件，包括 OpenStack 的和 ONOS。

资源地址	标题
https://github.com/opencord https://wiki.opencord.org/ https://wiki.opencord.org/ https://gerrit.opencord.org/#/admin/projects/ https://guide.opencord.org/overview.html https://www.opennetworking.org/cord/ https://guide.opencord.org/cord-5.0/profiles/ecord/	CORD

2.6 Cloudlet

Cloudlet，又称随我云（follow me cloud）、移动微云（mobile micro-cloud），派生于卡内基梅隆大学的 Elijah 项目，由 OpenEdgeComputing.org 组织推动，用于支持资源密集型和交互式的移动应用程序，满足增强现实应用、远程渲染的云游戏等低延迟、高带宽的需求。Cloudlet 主要包括四大特性：仅以软件形态部署、具备计算/连接/安全能力、就近部署、基于标准云技术构建等。

Cloudlets 是被视为 IoT 智能边缘计算策略的关键要素一部分，需要提供以下内容：

1) 低端到端应用程序延迟（实时）

2) 设备和本地“ cloudlet”之间的最大交互速率，以实现最佳计算结果（交互式）

3) 与专用网络的本地通信，以实现性能，隐私和安全性（安全）

4) 实时分析源数据，最小的云入口带宽（分析）

5) 使用动态过滤规则（分布式），在无线电局域网（RAN）中快速接入网络和其他功能。

Cloudlet 体系结构是一种参考体系结构，其目标是跨多个位置或位于同一位置的数据中心的联合云，参考如下图 5-2-16。该体系结构的目的是演示如何使用联合模式将多个分布式云连接在一起。该架构建立在 CAADE 架构之上

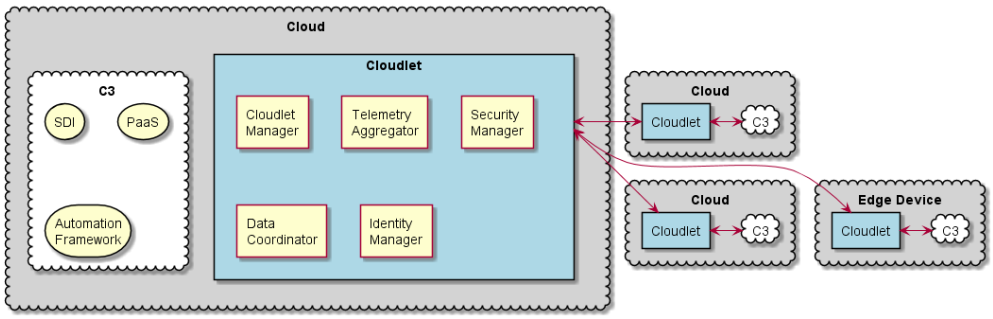

图 5-2-16 Cloudlet 架构

•信任管理器（Trust Manager）：管理多个跨数据中心的 TPM 中的安全密钥。

•Cloudlet 管理：每个云都有一个 Cloudlet 管理器，该管理器成为联合的一部分。

•数据协调器（Data Coordinator）：协调云之间的数据。

•联合编排云（Federated Orchestrated Cloud）：联合编排多个跨云调度服务请求。

•身份管理器（Identity Manager）：多个跨云管理身份

•遥测（Telemetry）：在将遥测共享给其他云之前聚合遥测。

•C3：通用混合云体系结构。 必须具备的关键要素，包括云管理平台，自动化框架和平台即服务框架。

Cloudlets 有两种主要的架构方法。第一个是基于标准星型模型的 Transient cloudlet（图 5-2-17），其中移动用户通过无线 LAN / RAN 访问附近的小云。瞬态云依赖于资源丰富的计算机基础结构，可通过无线网络（主要是蜂窝和 WLAN）提供移动设备可访问的数据存储和计算服务。

第二种是 Mobile cloudlet，其中一组资源丰富的移动设备设备（称为微云节点）可以在网状网络上相互连接并提供和使用服务。Mobile cloudlet 依靠对等网状通信，因此附近的一组移动设备可以通过安全的 Wi-Fi 或蓝牙进行连接。在此模型中，每个移动设备都利用分布式计算原理，将计算服务作为网格上的节点共享。

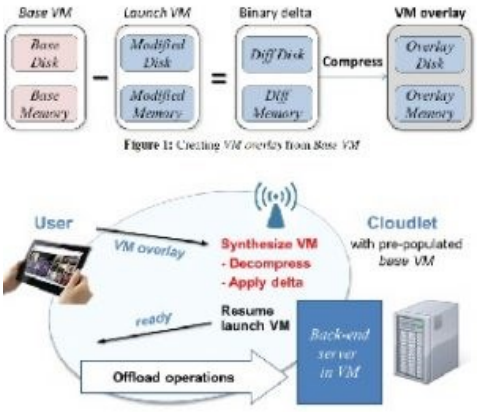

图 5-2-17 Transient cloudlet

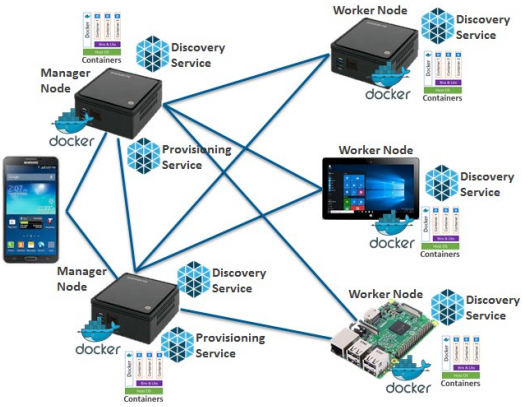

图 5-2-18 Mobile cloudlet

资源地址	标题
https://en.wikipedia.org/wiki/Cloudlet https://github.com/cmusatyalab/elijah-cloudlet https://github.com/cmusatyalab/elijah-provisioning http://elijah.cs.cmu.edu/ http://elijah.cs.cmu.edu/publications.html http://elijah.cs.cmu.edu/development.html https://www.openedgecomputing.org https://github.com/cmusatyalab/elijah-openstack https://github.com/CAADE	Cloudlet

2.7 ParaDrop

ParaDrop 是由威斯康星大学麦迪逊分校的 WiNGS 实验室开发的开源边缘计算平台。基于 Wi-Fi 路由器的边缘计算平台，用于管理边缘 Wi-Fi 接入点中运行的应用程序。目标是通过商用 Wi-Fi AP 硬件将网络智能带入家庭。灵活，安全，轻量级的容器虚拟化和多租户平台，允许基于网络的方法来实现与应用程序无关服务。

ParaDrop 平台特性：

1) 私密性。数据存储在网关的可用磁盘空间。

2) 低延迟。处理任务位于距离传感器只有一跳的网关进行。

3) 所有权友好。开发者可以将其程序打包、部署在网关，进而在虚拟环境中运行。

4) 本地网络环境。部分数据存储在网关，意味着只有终端用户请求其他的数据时才需要通过互联网途径发送到终端用户设备。

5) 允许离线。若与互联网的连接断开，基于云技术的传感器将失效，但家用网关始终开启这一特性，利用网关处理传感器中的数据，便能在断网情况下保证服务正常进行。

下图 5-2-21 给出了 ParaDrop 架构示意图，图中包括 ParaDrop 平台和两个示例应用程序。借助 ParaDrop API，第三方应用程序可以将服务部署到网络边缘-WiFi 路由器。

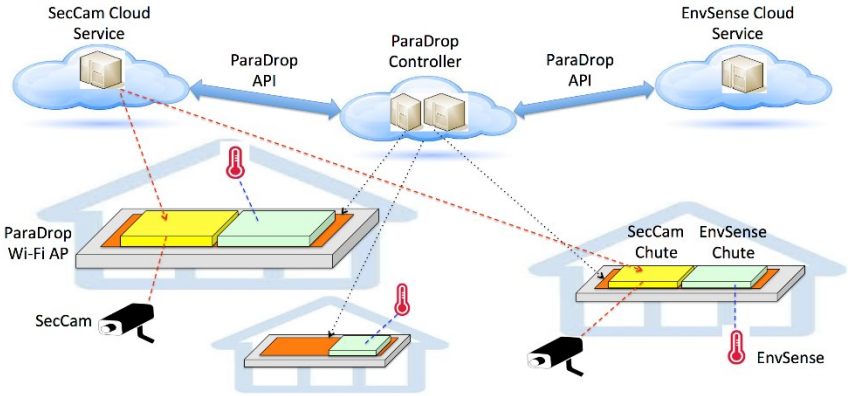

图 5-2-20 ParaDrop 架构

ParaDrop 具有 4 个主要组件：ParaDrop 边缘计算节点、ParaDrop 云控制器、ParaDrop 硬件、ParaDrop API。

ParaDrop 边缘计算节点：平台的定义组件（边缘计算节点）是运行应用程序和服务的计算平台。

ParaDrop 硬件： paradrop 软件平台可以在许多不同类型的硬件上运行，功能仅在具有无线网络接口的物理平台上可用。

ParaDrop 控制器：部署在 paradrop.org 上。用户可以通过网页注册以创建帐户。对于最终用户，它提供了一个仪表板来配置和监视他们拥有权限的 ParaDrop 路由器。还可以管理在 ParaDrop 路由器上运行的边缘计算服务（简称降落伞）。对于开发人员，它提供了一个接口，开发人员可以使用该接口注册其应用程序并管理插件。

ParaDrop API：通过 API 暴露平台的功能。根据功能和位置，API 可以分为两部分：云部分和边缘部分。云部分为应用程序提供管理界面，以从云中编排通道。包括资源许可管理，插件部署和管理，路由器配置管理等。边缘部分将路由器的本地上下文信息暴露到组件中。示例包括本地无线信道信息，本地无线外围设备访问等。

资源地址	标题
https://www.paradrop.org/ https://github.com/ParadropLabs https://paradrop.readthedocs.io/en/latest/ https://github.com/ParadropLabs/Paradrop https://wingslab.cs.wisc.edu	ParaDrop

2.8 Mbed

Arm Mbed OS 是 Arm 专门为物联网设备而设计的开源嵌入式操作系统，主要面向 ARM Cortex-M 系列微控制器，非常适合涉及智能城市、智能家庭和穿戴式设备等领域的应用程序。

相比于其他嵌入式操作系统，Arm Mbed OS 的主要优势在于：

1) 安全性：提供 mbed TLS 和 mbed uVisor 安全机制。

2) 连接性：支持多种协议栈，包括 Bluetooth LE, Wi-Fi, 6LoWPAN, Thread, Lora 等等。

3) 完整的工具链支持：提供在线 IDE, mbed CLI 以及第三方 IDE。

mbed 设备可通过以太网、WiFi 或低功耗蓝牙经，IPv6 或者 6LoWPAN 来连接。而 mbed OS 的安全性方面则采用了 uVisor，外加 TLS 和 DTLS 作为与外部设备和服务器的加密通信手段。这些设备遵从 LWM2M 或 CoAP 协议。

mbed 生态系统的各组成部分：

1) mbed 设备服务器（Device Server）：这是整个平台的核心组件，允许 web 应用连接和管理 mbed 设备。

2) mbed 设备连接器（Device Connector）：这是 mbed 设备服务器的托管版本，为开发者开发和测试他们的应用提供在线服务。

3) mbed 客户端（Client）：一套 C++ 库，用来从外部连接到 mbed 设备服务器或者设备连接器。

4) mbed TLS：加密解密库。

Mbed 软件体系架构分为应用层、中间层和硬件层，Mbed 主要实现了中间层功能部分功能，图下图 5-2-21 所示。

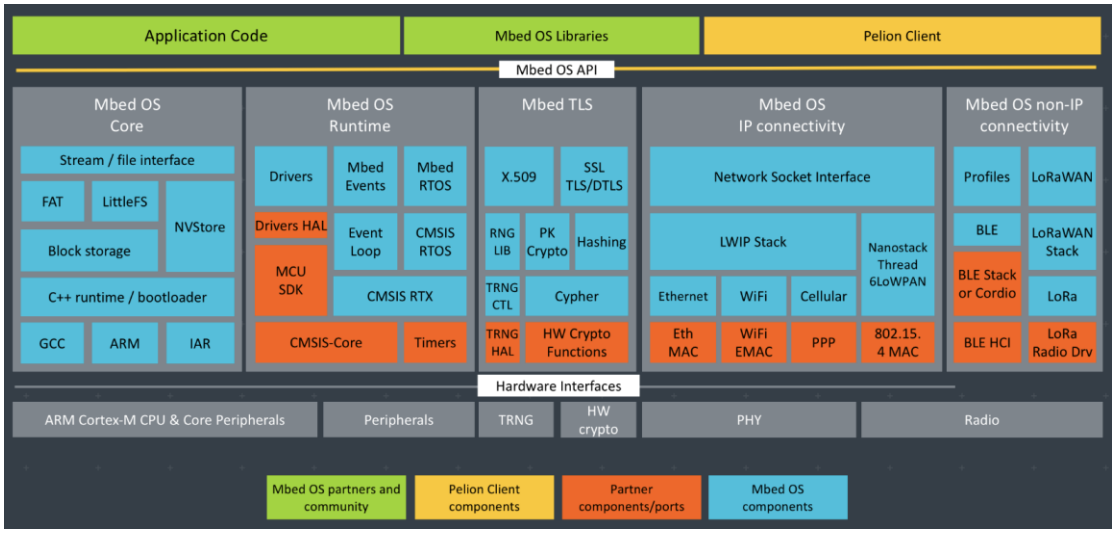

图 5-2-21 Mbed 板的基本架构

其他类似的 IoT 操作系统对比如下表：

OS	Brillo	mbedOS	RIOT	Contiki	Zephyr	Nuttx
Realtime	No	Yes	Yes	Yes	Yes	Yes
Footprint (RAM/ROM) *	35M RAM	32K/256K	~1.5K/~5K	< 10K/30K	8K RAM	16K/32K
Dev Language	C	C/C++	C/C++	C	C	C/C++
MCU w/o MMU	No	Yes	Yes	Yes	Yes	Yes
CPU core	Cortex-A	Cortex-M	Cortex-M, ARM7, x86	Cortex-M, x86	Cortex-M, x86	Cortex-A/M/R, x86, MIPS
Networking	Weave, 802.15.4 (Zigbee, Thread), BLE, WiFi, Ethernet	CoAP, BLE, 6LowPAN, Thread, WiFi, Ethernet, Cellular	CoAP, BLE, 6LowPAN, RPL, Ethernet	CoAP, 6LowPAN, RPL, Ethernet	CoAP, BT4.0, BLE, 6LowPAN, Ethernet	Ethernet
Cloud Service	Yes	Yes	No	No	No	No
Build System	repo/gcc	mbed/gcc (armcc)	gcc	gcc	kconfig/gcc	kconfig/gcc
License	Apache 2.0	Apache 2.0	LGPLv2	3-clause BSD	Apache 2.0	BSD
Repository	N/A	github	github	github	N/A	bitbucket

资源地址	标题
http://www2.keil.com/mdk5/cmsis/rtx https://www.mbed.com/en/ https://cloud.mbed.com/docs/current/introduction/index.html https://os.mbed.com/docs/v5.9/introduction/index.html https://github.com/ARMmbed https://www.mbed.com/zh-cn/development/getting-started/	ARM Mbed

2.9 其他资源

标题	简介	资源地址
Eclipse ioFog	IOT GateWay 的开发框架，虽然其使用 OSGI 作为支撑平台。可以在专用设备网络和本地网络，公共互联网或蜂窝网络之间的边界上生存，为该边界提供可管理和智能的网关，能够运行可以收集本地收集的信息并将其可靠传输的应用程序云端。	https://github.com/eclipse-iofog
Macchina.io	提供了一种“支持 Web、模块化、可扩展的”JavaScript 和 C++运行时环境，可用于开发在 Linux 开发板上运行的物联网网关应用程序。	https://github.com/macchina-io/macchina.io https://github.com/macchina-io/macchina.io/wiki
DSA	分布式服务架构(DSA)便于去中心化的设备互通、逻辑和应用程序。DSA 项目正在构建分布式服务链路(DSLinks)库，以便支持协议转换、与第三方数据源整合数据。	http://www.iot-dsa.org/
Project Flogo	Project Flogo 是一个轻量级的边缘计算平台，与任何特定的公共云平台无关	https://www.flogo.io/ https://github.com/TIBCOSoftware/flogo
Mirantis	MIRANTIS 云平台是一个综合的私有云软件栈，具有可靠、弹性、可扩展的特点，包括 openstack（裸机和虚拟机），kubernetes（容器），ceph（块存储和对象存储），opencontrail（面向 openstack 集群）和 Calico（面向 K8s） SDN。	https://www.mirantis.com/

ThingsBoard	ThingsBoard 是用于数据收集，处理，可视化和设备管理的开源物联网平台。它通过行业标准的物联网协议（MQTT，CoAP 和 HTTP）实现设备连接，并支持云和本地部署。	https://thingsboard.io/
SiteWhere	提供设备数据的摄取，存储，处理和集成的 IoT 平台。SiteWhere 运行在 Apache Tomcat 提供的核心服务器上。 它提供高度调整的 MongoDB 和 HBase 实现。	http://www.sitewhere.org/
DeviceHive	DeviceHive 是另一种功能丰富的开源 IoT 平台，它在 Apache 2.0 许可下分发。 DeviceHive 可以自由使用和更改。它提供了 Docker 和 Kubernetes 部署选项。	https://www.devicehive.com/
openVolcano	openVolcano 平台旨在在支持 5G 的基础架构中支持移动边缘和雾计算服务	http://openvolcano.org/ http://openvolcano.org/dokuwiki/doku.php?id=ov:download http://openvolcano.org/dokuwiki/doku.php?id=ov:installation
Kinetic Edge	边缘计算平台，它为云提供商，网络规模的公司和其他企业提供了最佳的方式来大规模交付边缘应用程序，而无需构建自己的数据中心或打乱自己的光纤	https://www.vapor.io/
OPENi	基于 Web 的开放源代码，应用程序与 Cloudlet 云的服务的集成框架	https://tssg.org/projects/openi/

2.10 参考文献

[1] Akraino Edge Cloud Stack[EB/OL]. https://wiki.akraino.org/display/AK/Akraino+Edge+Stack+Goal+and+Key+Principles

[2] GUPTA H, DASTJERDI A V, GHOSH S K, et al. iFogSim: a toolkit for modeling and simulation of resource management techniques in internet of things, edge and fog computing environments[J]. Software Practice & Experience, 2017, 47(9): 1275-1296.

[3] Baetyl 架构[EB/OL]. https://docs.baetyl.io/zh_CN/latest/overview/Design.html.

3 商业软件方案

标题	简介	资源地址
IBM Edge Computing	IBM Edge Computing 提供了更安全，自治的管理和治理功能，用于跨互联企业和 Industry 4.0 应用程序管理和扩展工作负载。提供了全面的部署灵活性，可以管理，移动和保护工作负载。可以将其部署为边缘服务器和网关上的完整或轻型 Kubernetes 集群，也可以将其部署在边缘设备上的轻量级 Docker 化容器中。	https://www.ibm.com/cloud/what-is-edge-computing
Google Cloud	云计算物联网边缘平台，扩展谷歌云计算的数据处理和机器学习边缘设备。	https://cloud.google.com/
Cisco IOx	Cisco IOx 应用程序环境结合了 Cisco IOS 和 Linux OS，可实现高度安全的联网。通过与 Cisco IOS 软件的安全连接，在雾中执行 IoT 应用，获得强大的服务，以实现与 IoT 传感器和云的快速可靠的集成。	https://www.cisco.com/c/en/us/products/cloud-systems-management/iox/index.html
GE Predix	GE 的 Predix 是一个基于 Cloud Foundry(CF)的云平台。主打工业物联网综合平台，致力于将设备端到云端打通，提供丰富的自有及第三方工业应用，进而把存储在云端的海量数据通过分析展现给使用者，三位一体提供一套完整的解决方案。	https://en.wikipedia.org/wiki/Predix_(software) https://www.ge.com/cn/b2b/digital/predix https://www.ge.com/digital/iiot-platform

NEV SDK	英特尔网络边缘虚拟化软件开发套件（NEVSDK）旨在协助应用程序开发者利用由 MEC 启用的机会，提供一个综合、的参考库和 API，NEV SDK 删除了需要应用程序开发人员了解复杂性和实现属性的底层移动网络协议。	https://www.intel.cn/content/www/cn/zh/communications/simplify-application-development-for-the-network-edge.html
IoTium	支持多租户，以允许多个云应用程序访问同一数据和设备。它还允许每个应用程序相互之间完全隔离，从而为 WAN 优化，QoS 和 Edge Analytics（分析）设置自己的策略和服务。	https://www.iotium.io/
AlefEdge	AlefEdge 软件位于 Open5G Edge Internet 的三大支柱的中心：边缘网关：移动边缘计算平台，提供边缘连接，超低延迟和支持边缘云的近乎计算能力 。边缘云：边缘云使计算，存储和 API 可以在应用程序的边缘本地可用。边缘应用程序：由于边缘技术，我们使新的和现有的应用程序具有空前的性能。	https://www.alefedge.com/
Edgeworx ioFog	Eclipse ioFog 软件版本，该版本使任何 Kubernetes 发行版都具有边缘感知能力，从而使客户能够创建真正的云到边缘连续体并部署应用程序和微服务从云到任何边缘设备。	https://projects.eclipse.org/projects/iot.iofog https://edgeworx.io/ https://edgeworx.io/kubernetes
MobiledgeX	MobiledgeX Edge-Cloud R1.0 按需部署边缘云位置（也称为“ cloudlets”）中的容器，以最佳方式满足所需应用程序和用户体验质量的需求。适用于 Java 和 C ++，C＃或 REST 的 Android 和 IOS 设备的与设备和平台无关的 SDK，支持边缘节点发现，内置身份和经过验证的位置服务，并能够自动连接到最近的边缘位置。	https://mobiledgex.com/ https://mobiledgex.com/technology/edge-cloud-r1-0
Qwilt	利用 5G 和 MEC 提供的前所未有的性能和速度，Qwilt 的云连接虚拟化 MEC 应用程序平台可确保您随时准备大规模，当今和未来交付最新，对延迟最敏感的应用程序。无论您是要实时传输实时流视频还是要为关键的物联网应用提供快速响应时间，Qwilt 的 Open Edge Cloud 都能提供最佳的用户体验。	https://qwilt.com/ https://wiki.opencaching.eu/index.php?title=Main_Page https://www.streamingvideoalliance.org/technical-groups/open-caching/ https://github.com/opencaching
Kinetic Edge	Vapor IO 开发了第一个端到端平台，用于在网络边缘构建和运行自主数据中心。称为 Kinetic Edge，Vapor IO 的技术使高度分布式的微数据中心可以嵌入无线和有线基础结构中，与最后一英里或无线电接入网（RAN）并置，并与软件和高速光纤啮合在一起用于远程操作，容错，低延迟工作负载和规模。	https://www.vapor.io/ https://www.vapor.io/kinetic-edge-alliance/ https://www.vapor.io/kinetic-edge/ https://hangar.com/kinetic-edge/

4 本章小结

5 参考文献

[1] 张建敏，谢伟良，杨峰义等，移动边缘计算技术及其本地分流方案[J]，电信科学 2016 年第 7 期。

[2] 张建敏，谢伟良，杨峰义等，基于 MEC 的 LTE 本地分流技术[J]，电信科学 2017 年第 6 期。

[3] 3GPP. Local IP access and selected IP traffic offload (LIPA-SIPTO) (release 10)：TR 23.829[S]. 2011.

[4] 3GPP. LIPA mobility and SIPTO at the local network (release 11): TR 23.859[S]. 2011.

[5] V. Miliotis, L. Alonso, C. Verikoukis, “Energy efficient proportionally fair uplink offloading for IP flow mobility,” Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp. 6–10, 2014.