

Lec14 - IoT平台关键技术

设备管理、边缘计算、通信协议



设备管理

什么是设备管理？

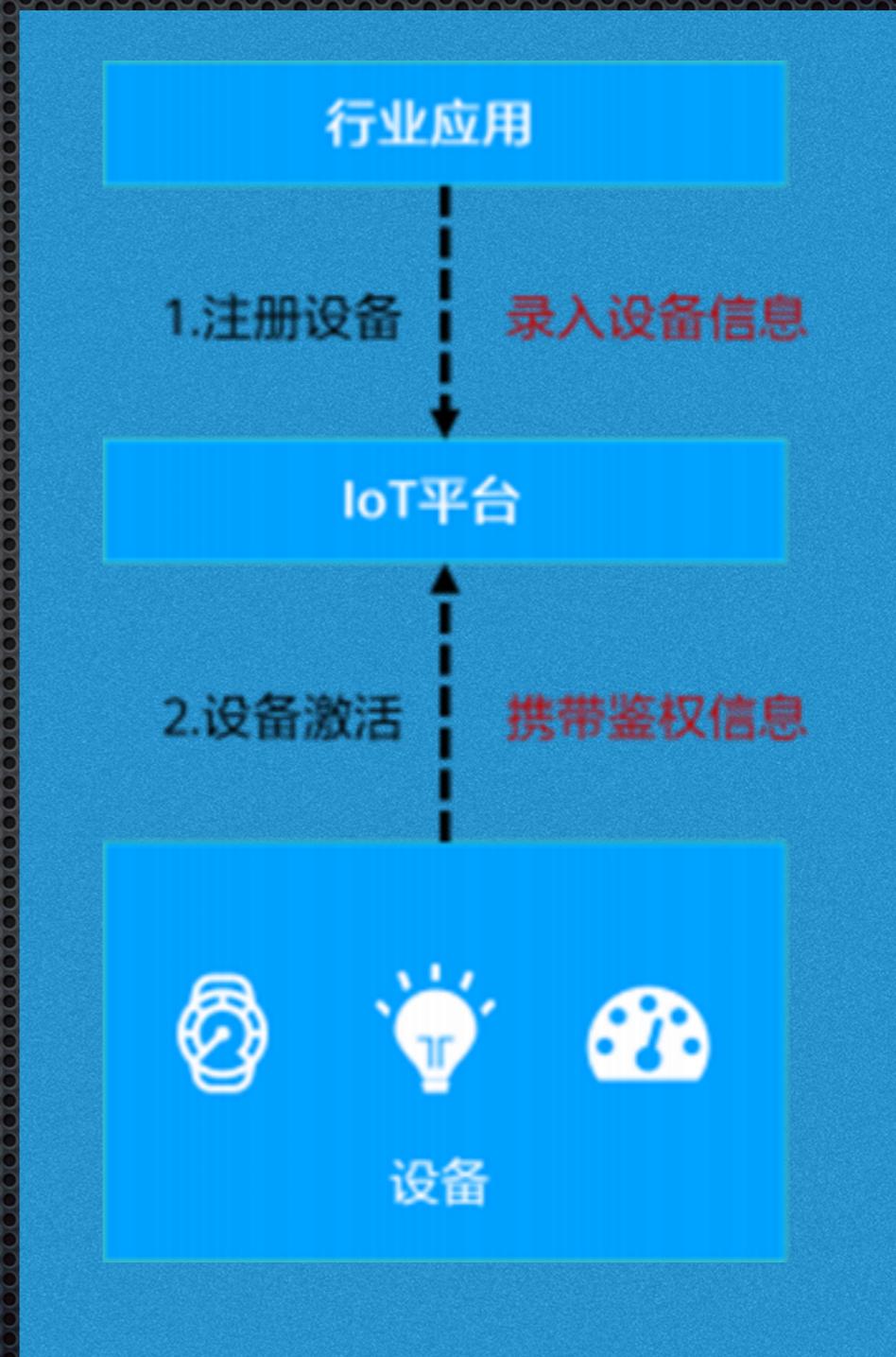
- 设备管理服务, 可以帮助对所有连接的设备, 在全球范围内进行规模化的注册、查看及远程管理
 - 可以在 IoT 平台内查看到所有设备队列, 并进行信息跟踪、运行状态查看、管理连接、查看日志数据等动作
 - 通过固件标准上报的版本信息, 还可以定位到每个设备当下运行的固件信息, 便于确认设备的稳定并排查问题
 - 可以通过无线下载技术 (OTA), 发送固件升级, 从而确保设备始终运行最新版本软件
- 设备管理在设备接入基础上, 提供了更丰富完备的设备管理能力, 简化海量设备管理复杂性, 节省人工操作, 提升管理效率

主要功能

- 物联网平台提供功能丰富的设备管理服务，包括不限于：
 - 生命周期
 - 设备分组
 - 设备影子
 - 物模型
 - 数据解析
 - 数据存储
 - 在线调试
 - 固件升级
 - 远程配置
 - 实时监控等

设备注册&接入鉴权

- 为了保证接入物联网平台的设备是安全可信的，需要进行设备注册和设备接入鉴权操作
 - 设备注册，指用户通过控制台或调用注册设备API在IoT平台中注册设备信息，平台中存在设备信息后，再接入真实的实体设备，这样平台与终端实体设备可以实现连接和通信
 - 设备接入鉴权，是指IoT平台对接入平台的设备进行鉴权认证，用于保障设备接入信息的完整性和安全性、设备与平台消息传输完整性和安全性



命令下发

- 设备的产品模型中定义了IoT平台可向设备下发的命令，平台向设备下发命令，修改设备的服务属性，实现对设备的控制
- 平台向设备下发命令包括立即下发和缓存下发两种情况，如下

命令下发机制	定义	适用场景	NB-IoT设备
立即下发	不管设备是否在线，平台收到命令后立即下发给设备。如果设备不在线或者设备没收到命令则下发失败。支持给本应用的设备和被授予权限的其他应用的设备下发命令。	立即下发适合对命令实时性有要求的场景，如路灯开关灯，燃气表开关阀。使用立即下发时，命令下发的时机需要由应用服务器来确定。	适用，需将省电模式设置为DRX或eDRX模式，且需要在物联网平台与EPC网络之间建立IPSEC隧道。
缓存下发	物联网平台在收到命令后先缓存，等设备上线或者设备上报数据时再下发给设备，如果单个设备存在多条缓存命令，则进行排队串行下发。支持给本应用的设备和被授予权限的其他应用的设备下发命令。	缓存下发适合对命令实时性要求不高的场景，如配置水表参数，	适用，需将省电模式设置为PSM模式。

设备联动规则

- 规则引擎的使用对象是终端用户，系统已经预置支持的规则场景，终端用户通过方便、易理解的友好界面在自有设备下制定自动化规则
- 规则可以和设备、应用、告警绑定，当绑定的信息满足条件时，规则可以自动化执行响应动作
- 规则引擎定位处理各种事件，利用规则引擎可以完成异常事件的及时通知和快速处理，帮助终端用户维护设备、监控设备，保证系统业务的及时恢复
- 阈值超限、范围超限、位置跟踪等事件，也可定义为规则引擎输入条件，并关联对应的处理动作

设备联动规则示例



设备批操作

- 设备批操作向物联网平台提供对终端设备统一管理的通道，能提升对终端设备的管理效率，很好地满足用户批量管理设备的需求
- 当用户接入的设备数量过多，或者需要对全部或某一个群组的设备进行相同的操作时，可以采用设备批操作
- 物联网平台支持对设备的批量操作，包括批量注册设备、批量命令下发、批量位置上传、批量设备配置和批量软固件升级
 - 批量注册设备：因注册设备数量过多而导致注册时间太长，可采用批量注册的方式注册设备
 - 批量命令下发：当物联网平台需要对批量设备下发命令时，可通过北向接口创建批量命令下发任务，在物联网平台上，可查看任务的执行状态、操作者和成功率等信息
 - 批量位置上传：当物联网平台需要对设备的位置进行批量上传时，可采用批量位置上传操作。这里的设备主要指安装位置相对固定的终端设备，如水表
 - 批量设备配置：当物联网平台需要对设备进行批量配置时，可采用批量设备配置操作
 - 批量软固件升级：当物联网平台需要对设备的固件或者软件进行批量升级时，可采用批量软固件升级操作

设备管理—物模型

- 物模型（Thing Model）是对设备在云端的功能描述，包括设备的属性、数据、服务和事件
- 物联网平台通过定义一种物的描述语言来描述物模型，称之为 TSL（即 Thing Specification Language），采用JSON格式，可以根据TSL组装上报设备的数据

物模型的定义-阿里

功能类型

说明

属性 (Property)

一般用于描述设备运行时的状态，如环境监测设备所读取的当前环境温度等。属性支持GET和SET请求方式。应用系统可发起对属性的读取和设置请求。

服务 (Service)

设备可被外部调用的能力或方法，可设置输入参数和输出参数。相比于属性，服务可通过一条指令实现更复杂的业务逻辑，如执行某项特定的任务。

事件 (Event)

设备运行时的事件。事件一般包含需要被外部感知和处理的通知信息，可包含多个输出参数。如，某项任务完成的信息，或者设备发生故障或告警时的温度等，事件可以被订阅和推送。

是什么

做什么

可对外
提供什么

<https://www.alibabacloud.com/help/zh/doc-detail/73727.htm>

```
{  
    "schema": "物模型结构定义的访问URL",  
    "profile": {  
        "productKey": "产品ProductKey"  
    },  
    "properties": [  
        {  
            "identifier": "属性唯一标识符 (产品下唯一)",  
            "name": "属性名称",  
            "accessMode": "属性读写类型: 只读 (r) 或读写 (rw)。",  
            "required": "是否是标准功能的必选属性",  
            "dataType": {  
                "type": "属性类型: int (原生)、float (原生)、。。。。。",  
                "specs": {  
                    "min": "参数最小值 (int、float、double类型特有)",  
                    "max": "参数最大值 (int、float、double类型特有)",  
                    "。。。。。 } } }  
    ],  
    "events": [  
        {  
            "identifier": "事件唯一标识符 (产品下唯一, 其中post是默认生成的属性上报事件。)",  
            "name": "事件名称",  
            "desc": "事件描述",  
            "type": "事件类型 (info、alert、error)",  
            "required": "是否是标准功能的必选事件",  
            "outputData": [],  
            "method": "事件对应的方法名称 (根据identifier生成)"  
        }  
    ],  
    "services": [  
        {  
            "identifier": "服务唯一标识符",  
            "name": "服务名称",  
            "desc": "服务描述",  
            "required": "是否是标准功能的必选服务",  
            "callType": "async (异步调用) 或sync (同步调用)",  
            "inputData": [],  
            "outputData": [],  
            "method": "服务对应的方法名称 (根据identifier生成)"  
        }  
    ]  
}
```

设备影子

- 物联网平台提供设备影子功能，用于缓存设备状态
 - 设备在线时，可以直接获取云端指令
 - 设备离线后，再次上线可以主动拉取云端指令
- 设备影子是一个 JSON 文档
 - 存储设备上报状态
 - 应用程序期望状态信息
- 每个设备有且只有一个设备影子，设备可以通过MQTT获取和设置设备影子来同步状态
 - 该同步可以是影子同步给设备，也可以是设备同步给影子

应用场景

- 场景1:
 - 网络不稳定，设备频繁上下线
- 场景2:
 - 多程序同时请求获取设备状态
- 场景3:
 - 设备掉线
 - 指令携带时间戳

```
{  
  "state": {  
    "desired": {  
      "color": "RED",  
      "sequence": [  
        "RED",  
        "GREEN",  
        "BLUE"  
      ]  
    },  
    "reported": {  
      "color": "GREEN"  
    }  
  },  
  "metadata": {  
    "desired": {  
      "color": {  
        "timestamp": 1469564492  
      },  
      "sequence": {  
        "timestamp": 1469564492  
      }  
    },  
    "reported": {  
      "color": {  
        "timestamp": 1469564492  
      }  
    }  
  },  
  "timestamp": 1469564492,  
  "version": 1  
}
```

关于模板

- 模板定义了一类设备的schema， Schema中定义了各设备各属性的显示名称、类型、默认值等信息
 - 通过模板，我们定义了设备的视图(view)
 - 此外，需要支持模板的CRUD操作， IoT Device Management Schema API 主要包含管理设备模板的相关接口--<https://cloud.baidu.com/doc/IOT/s/Bjwvy7j1b>
- 例如：
 - 物模型模板--<https://open.iot.10086.cn/doc/oes/book/ecp/objectmodelmanagement.html>
 - Device template library in IoT Central--<https://azure.microsoft.com/en-us/blog/device-template-library-in-iot-central/>

数字孪生

- 数字孪生，英文名叫Digital Twin（数字双胞胎），也被称为数字映射、数字镜像
- 起源
 - 工程实践：
 - 2009年，AFRL（美国空军研究实验室）发起了一个“**机身数字孪生**”项目
 - NASA，2010年首次提出了数字孪生（Digital Twins）的概念
 - 更有理论色彩：
 - 2002年Michael Grieves在密歇根大学提出了PLM（产品生命周期管理）概念模型
- 数字孪生的概念，起源于制造业，现在已广泛应用到了智慧城市、智慧交通、智慧农业、智慧医疗、智能家居等行业。简而言之，数字孪生无处不在。



图 2 数字孪生行业应用

数字孪生

- 官方定义：
 - 数字孪生，是充分利用物理模型、传感器更新、运行历史等数据，集成多学科、多物理量、多尺度、多概率的仿真过程，在虚拟空间中完成映射，从而反映相对应的实体装备的全生命周期过程

数字孪生的价值

- 数字孪生的概念非常大。简单的看一下这个概念所提供的典型的好处：
 - 可见性：数字孪生能够实现机器操作的可见性，以及制造工厂或者机场中大型的互联系统的可见性
 - 预测性：使用多种建模技术(基于物理和基于数学的)，数字孪生模型能够用于预测机器未来的状态
 - 假设分析：通过适当设计的接口，可以很容易的与模型进行交互，并且对模型询问假设问题，来模拟现实中无法创建的各种条件
 - 对行为进行理解和解释的记录与沟通机制：数字孪生模型能够作为一种沟通和记录机制，能够对单独的机器或者机器的集合的行为进行理解和解释
 - 连接不同的系统：比如后端的业务应用
 - 如果设计的正确，数字孪生模型能够用来连接后端的业务应用，在供应链运作中实现业务成果，包括制造、采购、仓储、运输、物流、现场服务等

数字孪生与仿真(Simulation)的区别

■ 仿真技术

- 应用仿真硬件和仿真软件通过仿真实验，借助某些数值计算和问题求解，反映系统行为或过程的模型技术
- 是将包含了确定性规律和完整机理的模型转化成软件的方式来模拟物理世界的方法
- 目的是依靠正确的模型和完整的信息、环境数据，反映物理世界的特性和参数
- 仿真技术仅仅能以离线的方式模拟物理世界，不具备分析优化功能，因此不具备数字孪生的实时性、闭环性等特征

■ 数字孪生

- 需要依靠包括仿真、实测、数据分析在内的手段对物理实体状态进行感知、诊断和预测，进而优化物理实体，同时进化自身的数字模型
- 仿真技术作为创建和运行数字孪生的核心技术，是数字孪生实现数据交互与融合的基础
- 在此基础之上，数字孪生必需依托并集成其他新技术，与传感器共同在线以保证其保真性、实时性与闭环性

数字孪生与信息物理系统(CPS)的异同

- 数字孪生与 CPS 都是利用数字化手段构建系统为现实服务
- 不同：
 - CPS 属于系统实现，而数字孪生侧重于模型的构建等技术实现
 - 相比于综合了计算、网络、物理环境的多维复杂系统 CPS，数字孪生的构建作为建设 CPS 系统的使能技术基础，是 CPS 具体的物化体现

数字孪生与元宇宙

- 通过虚拟增强的物理现实，呈现收敛性和物理持久性特征的，基于未来互联网的，具有连接感知和共享特征的3D虚拟空间
- 共同点：
 - 都以数字技术为基础，再造高仿真的数字对象和事件，以进行可视化感知交互和运行
 - 底层支撑技术可通用
- 不同的是
 - 元宇宙可以现实物理世界为数字框架，也可以完全塑造全新的理念数字世界，终极形态是基于数字世界实现的原生社会
 - 而数字孪生则是以信息世界严格、精确映射物理世界和事件过程为框架和基础，无论是工业制造，还是城市管理，基于实时客观数据的动态进程，与人工智能结合的挖掘分析和深度学习；并进一步模拟情境和决策，以改进现实或更好适应现实，最终实现自动控制或自主决策控制，终极形态是自主孪生

行业实现

1. 简单的设备模型

- 这些实现通常使用包含两个主要属性集的JSON文档：
 - 一组观测值或者报告值：通常，设备上的探测器读取当前值，并更新这些观测属性。比如，一台机器当前观察的转速(比如 1000 RPM)
 - 一组期望值：这是控制程序希望在设备上设置的值。比如，一个应用能够设置引擎转速到1200 RPM
- 除了这两组主要的属性，这些实现也在JSON文档中存储了相关的信息，比如设备的名字或者序列号，或者当前位置



行业实现

2. 工业孪生

- 这类实现通常被工业物联网供应商所采用，它包括PLM工具设计机器的信息和一台设备的模型，一些工业供应商关注物理属性、设计信息和实时数据，并且将他们展现在一个资产/设备模型图当中
- 值得注意的是，这些模型通常都基于机器的物理属性

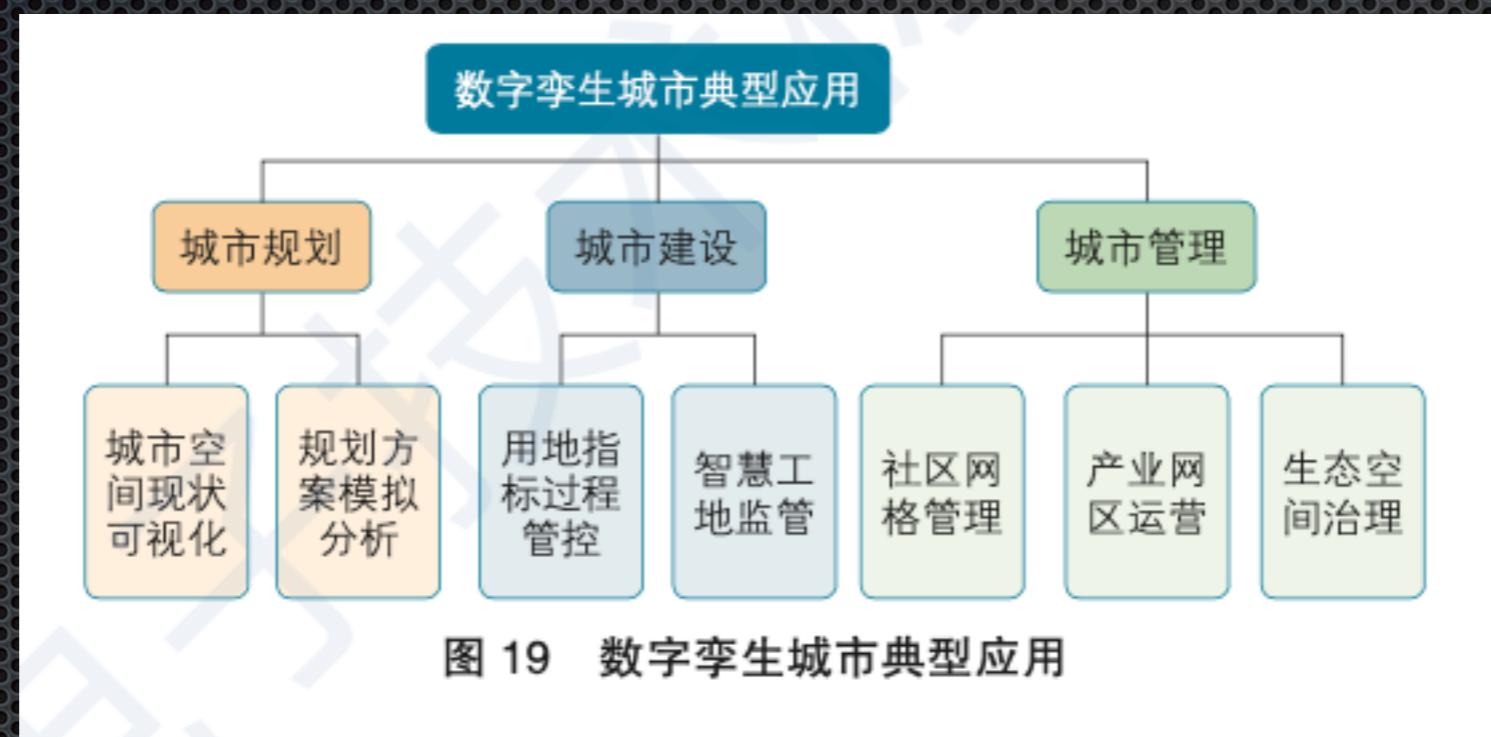


智慧城市领域数字孪生应用

- 2019 年中国新型智慧城市规模超过 9000 亿元，未来几年将保持较快速度增长，预计到 2023 年市场规模将超过 1.3 万亿元
- 当前，安全综治、智慧园区、智慧交通是智慧城市建设投入的重点，三大细分场景规模占智慧城市建设总规模的 71%，而城市级平台、机器人等新技术和产品则在快速落地，被更多城市建设方采纳和应用



典型应用场景介绍





边缘计算

基于云的IOT解决方案不足

- 首先，对于大规模边缘的多源异构数据处理要求，无法在集中式计算线性增长的计算能力下得到满足
 - IOT感知层是海量数据，数据具有很强的冗余性、相关性、实时性和多源异构性，数据之间存在着频繁的冲突与合作
 - 需要实时处理
- 其次，数据在用户和云数据中心之间的长距离传输将导致高网络时延和计算资源浪费
- 再次，大多数终端用户处于网络边缘，通常使用的是资源有限的移动设备，具有低存储和计算能力以及有限的电池容量，所以有必要将一些不需要长距离传输到云数据中心的任务分摊到网络边缘端
- 最后，云计算中数据安全性和隐私保护在远程传输和外包机制中将面临很大的挑战，使用边缘计算处理数据则可以降低隐私泄漏的风险

边缘计算发展

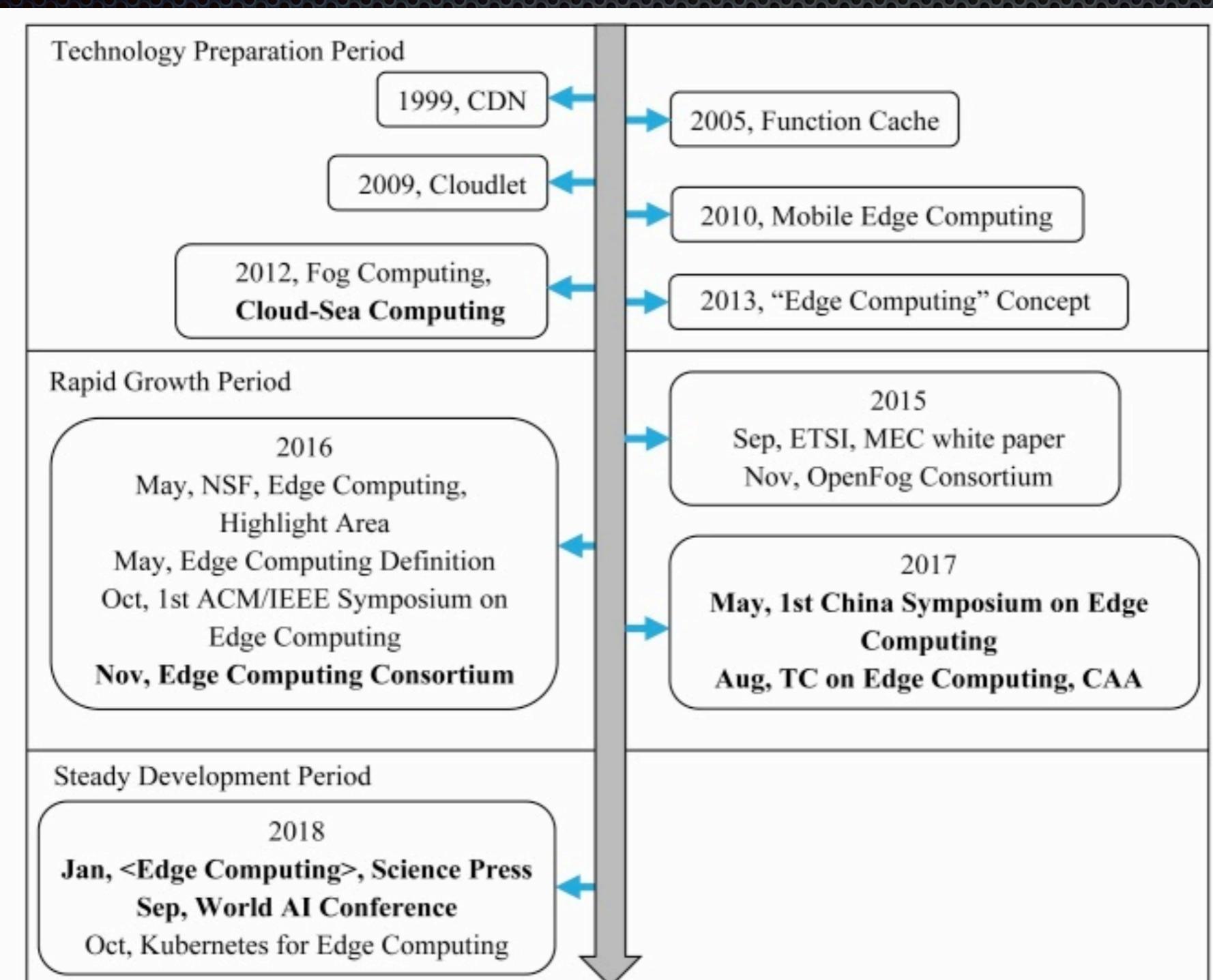
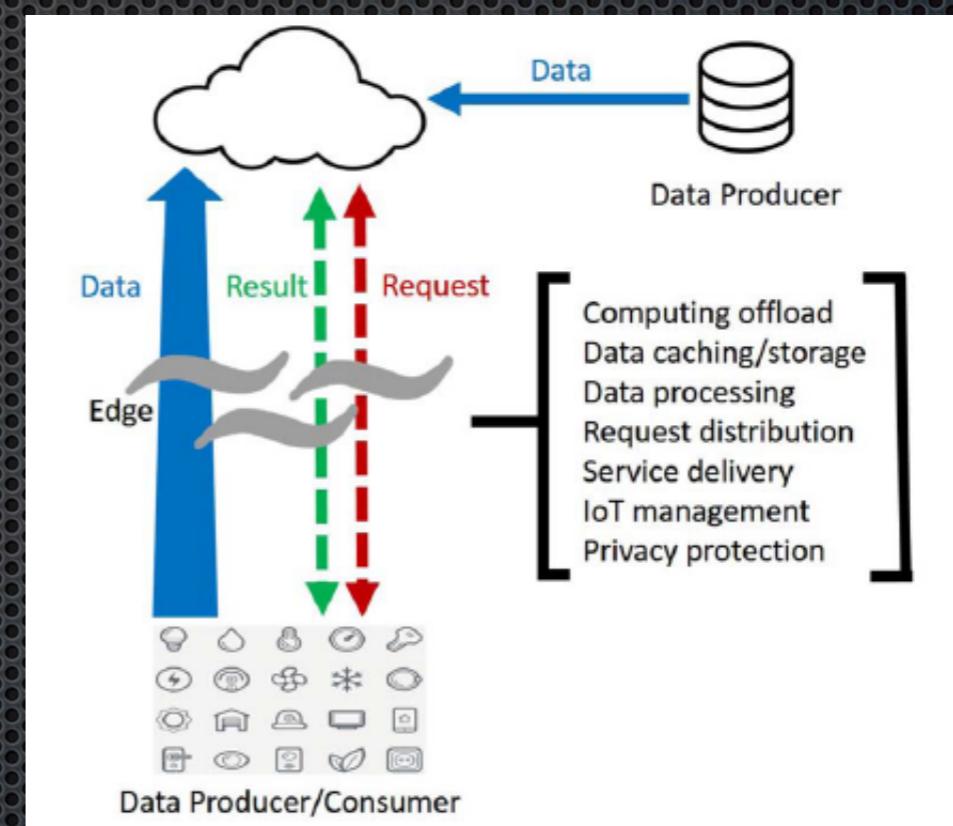


Fig. 2 Development states of edge computing and the typical events at each stage

图 2 边缘计算的发展历程及典型事件

边缘计算是什么？

- 我们将“边缘”定义为数据源和云数据中心之间的任何计算和网络资源
- 边缘计算与雾计算是可以互换的

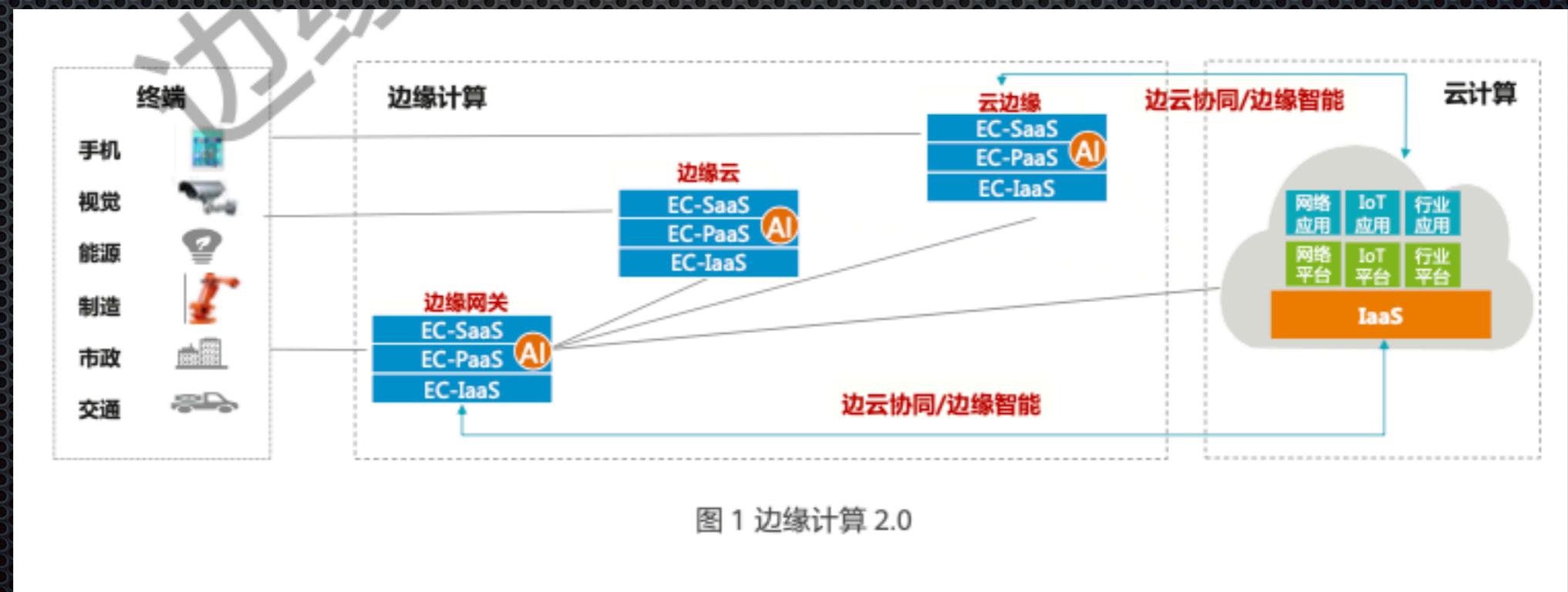


边缘计算的定义

- ISO/IEC JTC1/SC38：边缘计算是一种将主要处理和数据存储放在网络的边缘节点的分布式计算形式
- 国际标准组织ETSI的定义为在移动网络边缘提供IT服务环境和计算能力，强调靠近移动用户，以减少网络操作和服务交付的时延，提高用户体验

边缘计算2.0

- 主要包括云边缘、边缘云和边缘网关三类落地形态；以“边云协同”和“边缘智能”为核能力发展方向；
- 软件平台需要考虑导入云理念、云架构、云技术，提供端到端实时、协同式智能、可信赖、可动态重置等能力；
- 硬件平台需要考虑异构计算能力，如鲲鹏、ARM、X86、GPU、NPU、FPGA等。

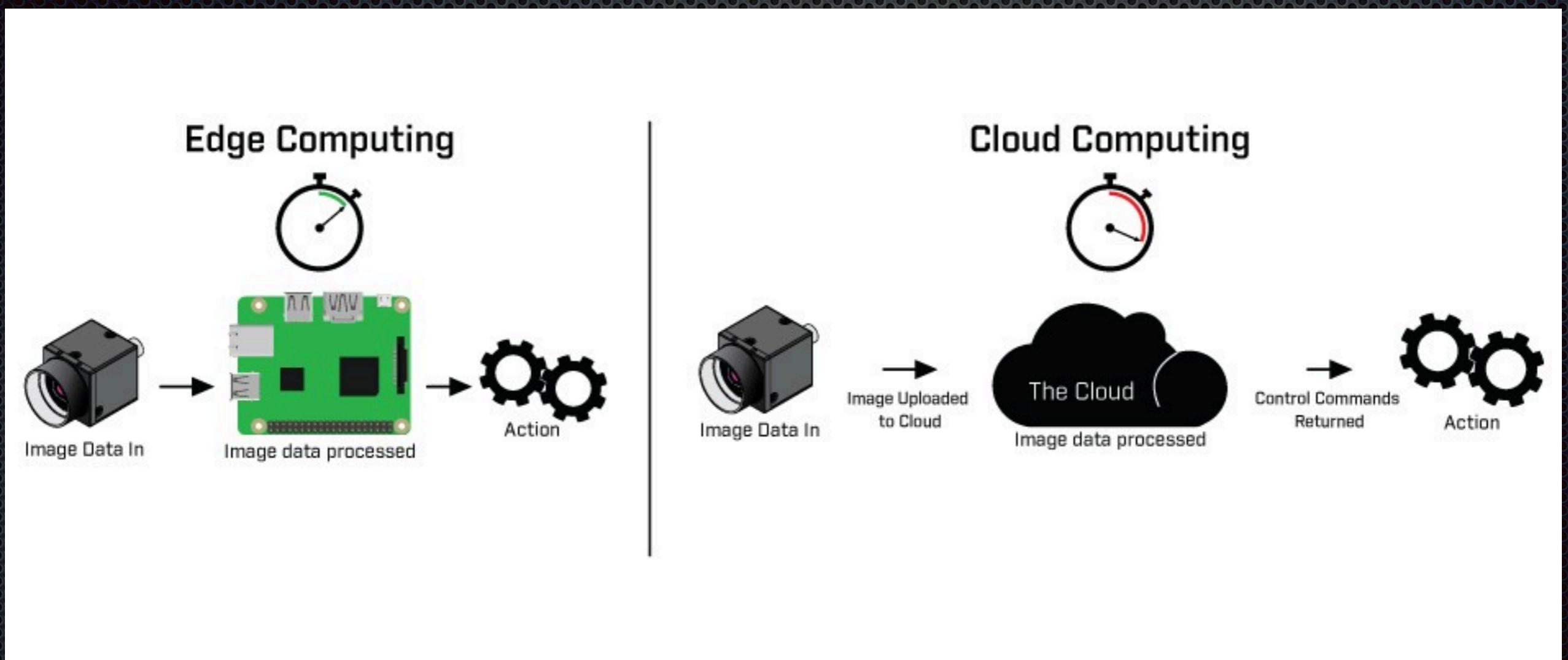


特征	Cloud	Edge/Fog
位置	集中	分布在不同的地理位置
容量	非常大的数据中心	许多小规模的雾节点，形成一个大系统
能耗	高	低
延迟	高，因为终端用户和云之间的距离很大	低，因为边缘和最终用户之间的距离很短
资源和服务的邻近性	远离终端用户，在数据中心	靠近终端用户，处于网络边缘
应用	支持不需要短延迟的应用程序，主流云应用	支持大多数类型的应用程序，如VR、智能家居、智能车辆、智慧城市
服务成本	高，因为数据中心被大公司垄断	由于在网络边缘处理数据，成本更低

边云协同放大各自价值

- 边缘计算的CROSS(Connectivity连接、Realtime实时、Optimization数据优化、Smart智能、Security安全)价值推动计算模型从集中式的云计算走向更加分布式的边缘计算
- 边缘计算与云计算各有所长
 - 云计算擅长全局性、非实时、长周期的大数据处理与分析，能够在长周期维护、业务决策支撑等领域发挥优势
 - 边缘计算更适用局部性、实时、短周期数据的处理与分析，能更好地支撑本地业务的实时智能化决策与执行
- 因此，边缘计算与云计算之间不是替代关系，而是互补协同关系

Edge computing



*Figure 1: Edge computing processes image data close to the source for low system latency
Figure 2: Cloud computing results in a long signal path for image data which increases system latency*

边缘计算例子

- 边缘计算是一种网络模型，其中数据处理发生在网络的边缘，靠近数据源。边缘计算可以消除将图像数据发送到中央服务器或云服务进行处理的需要。
- 例如，用于道路收费的边缘计算将使系统能够在靠近摄像头的低功耗单板计算机上进行车牌识别。只会传输车牌号码，而不会传输车辆或道路的完整图像。

边缘计算的主要优势

减少带宽

在源头处理数据，消除了将图像传输回中央服务器的需要。由于只发送可操作的信息，因此所需的带宽要少得多。

减少延迟

减少从边缘发送的数据量可以加快系统速度，并最大限度地减少捕获图像和到达信息之间的延迟。

改善隐私提升安全性

车牌、人脸等敏感信息不会传输到云端。

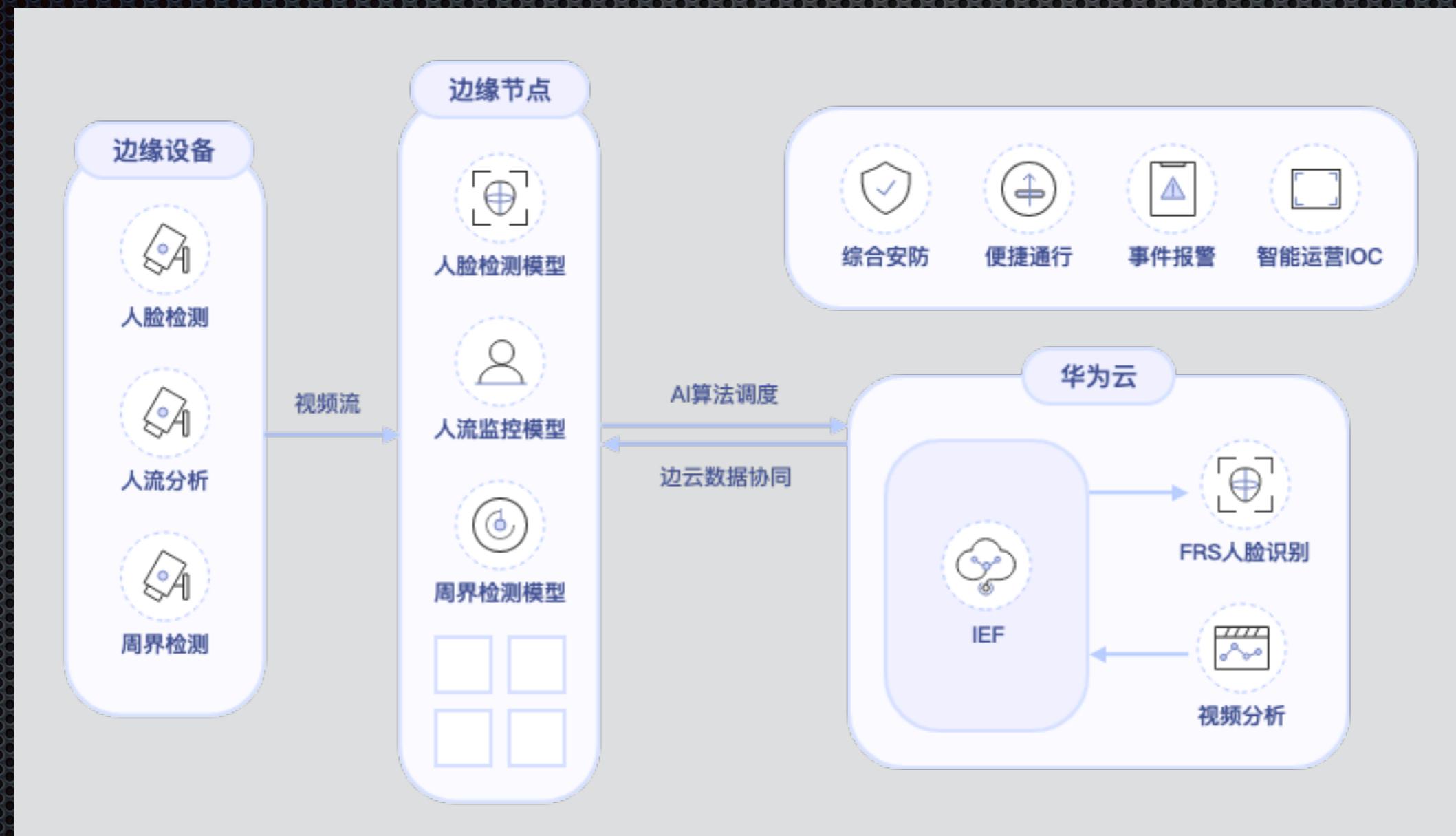
何时使用边缘计算

应用	优势
智能交通系统	更低的带宽消耗，提高系统安全性，降低隐私风险
工业自动化	更低的延迟和抖动，以实现更高的吞吐量
自动驾驶车辆导航	最大限度地减少系统延迟，使高速车辆能够快速做出决策，同时消除需要始终在线的数据连接的依赖

华为车路协同解决方案架构



监控场景（华为）



<https://www.huaweicloud.com/product/ief.html>

IoT解决方案

云



边缘

端

面向物联网端的边缘计算开源平台

- 面向物联网端的边缘计算开源平台，致力于解决在开发和部署物联网应用的过程中存在的问题，例如设备接入方式多样性问题等
- 这些平台部署于网关、路由器和交换机等边缘设备，为物联网边缘计算应用提供支持
- 代表性的平台是Linux基金会发布的 EdgeXFoundry 和 Apache 软件基金会的 Apache Edgent

面向物联网端的边缘计算开源平台

- EdgeXFoundry的架构图



服务层描述

- 设备服务层。设备服务层主要提供设备接入的功能，由多个设备服务组成。每个设备服务是用户根据设备服务软件开发工具包（SDK）编写生成的一个微服务。EdgeX Foundry 使用设备文件去定义一个南侧设备的相关信息，包括源数据格式，存储在 EdgeX Foundry 中的数据格式以及对该设备的操作命令等信息。设备服务将来自设备的数据进行格式转换，并发送至核心服务层。目前，EdgeX Foundry 提供了消息队列遥测传输协议（MQTT）、ModBus 串行通信协议和低功耗蓝牙协议（BLE）等多种接入方式。
- 核心服务层。核心服务层由核心数据、命令、元数据、注册表和配置 4 个微服务组件组成。核心数据微服务存储和管理来自南侧设备的数据、元数据微服务存储和管理设备的元数据。命令微服务将定义在设备文件的操作命令转换成通用的 API，提供给用户以监测控制该设备。注册表和配置微服务存储设备服务的相关信息。
- 支持服务层。支持服务层提供边缘分析和智能服务，以规则引擎微服务为例，允许用户设定一些规则，当检测到数据满足规则要求时，将触发一个特定的操作。例如规则引擎可监测控制温度传感器，当检测到温度低于 25 度时，触发对空调的关闭操作。
- 导出服务层。导出服务层用于将数据传输至云计算中心，由客户端注册和分发等微服务组件组成。前者记录已注册的后端系统的相关信息，后者将对应数据从核心服务层导出至指定客户端。
- 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。



IOT通信协议

IOT协议

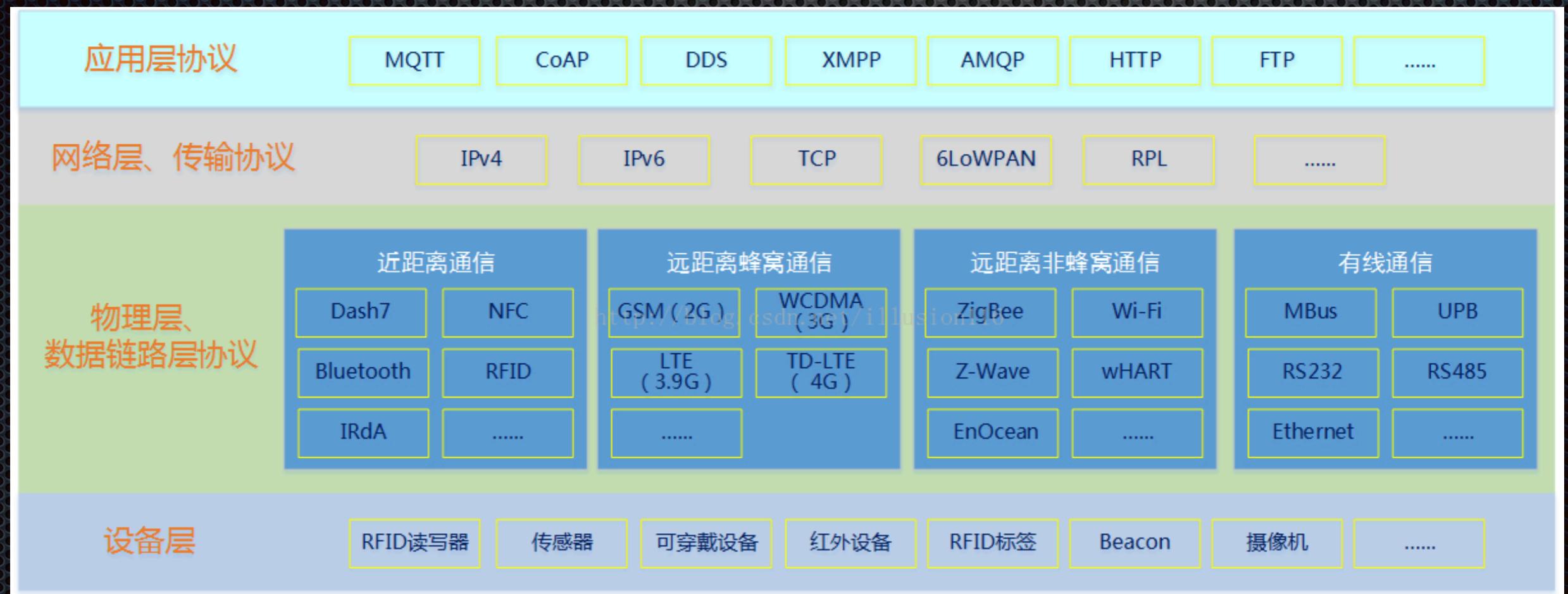


TABLE 1. Some of the data link layer protocols comparison.

Technology	Throughput (Approximate) (Kbps)	Range (Approximate) (m)	Mobility Support
NFC	424	0.1	Yes
ANT+	1,000	50	Yes
ZigBee	250	100	Yes
Z-Wave ¹	40	100	Yes
Bluetooth ³	1,000	100	Yes
WiFi	54,000	150	Yes
WirelessHART	250	150	Yes
Weightless-W	10,000	5,000	Yes
LTE-M	1,000	11,000	Yes
LoRaWAN	0.3 ⁴	14,000	Yes
Sigfox ²	0.1	17,000	Yes
NB-IoT	200	20,000	No ⁵

¹ outdoor or open air; indoor is approximately 50m

² data rate may vary depending on the deployed region (up to 600 bps)

³ Bluetooth 5 can support a range of approximately 150m (outdoor) with up to 8x broadcasting capacity

⁴ range up to 50kbps if using Frequency-Shift Keying (FSK) instead of LoRa

⁵ minimal, no full support for mobility as in LTE (possibly during cell reselection - idle state)

LINK LAYER PROTOCOLS

- 802.3 – Ethernet
- 802.11 – wifi
- 802.16 – WiMax
- 802.15.4 – Low Rate WPAN
- 2G/3G/4G/5G – Mobile Communication

NETWORK LAYER PROTOCOLS

- IPv4
 - Exhausted in 2011
 - 32bit address
- 128 bit addresses 6LoWPAN
 - Limited processing capability
 - Shows compression mechanism with IPv6 over 802.15.4

TRANSPORT LAYER PROTOCOL

- TCP
- UDP

APPLICATION LAYER PROTOCOL

- HTTP/REST
- 受限应用协议(CONSTRAINED APPLICATION PROTOCOL, CoAP)
- 消息队列遥测传输 (Message Queue Telemetry Transport, MQTT)
- 高级消息队列协议 (ADVANCED MESSAGE QUEUING PROTOCOL, AMQP)
- 数据分发服务DDS (DataDistributionService, DDS)
- 可扩展通讯和表示协议 (Extensible Messaging and Presence Protocol, XMPP)
- WebSocket

IoT APPLICATION RANGE REQUIREMENTS

TABLE 2. IoT application range requirements [120]–[123], [130]–[140].

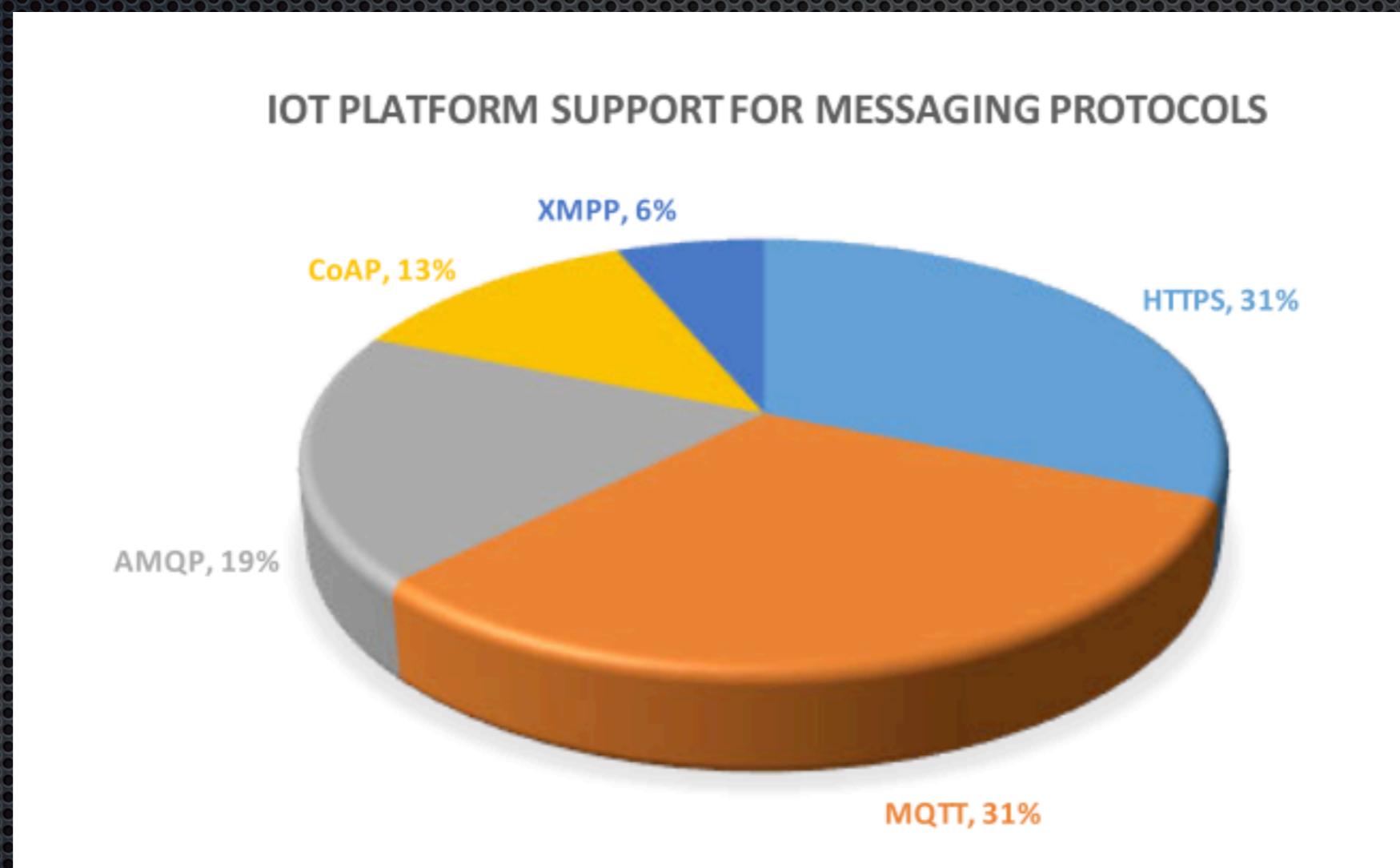
Application	~ Range	Technology
Industry Automation	10m - 50m	LoRa, ZigBee, WirelessHART
Smart Metering	15km - 40km	LoRa, Weightless-N
Smart Buildings	10m - 250m	LoRa, Sigfox
Asset Tracking	50m - 500m	LoRa, Sigfox, Weightless
Smart Energy	100m - 15km	LoRa
Environmental Monitoring	100m - 1.5km	LoRa, Sigfox
Health Monitoring	10m - 25m	BLE, LoRa, ZigBee, ANT+
Wearable & Fitness	30m-50m	ANT+, BLE
Consumer Electronics	10m-25m	ZigBee, Z-Wave, BLE

云上支持的物联网消息协议

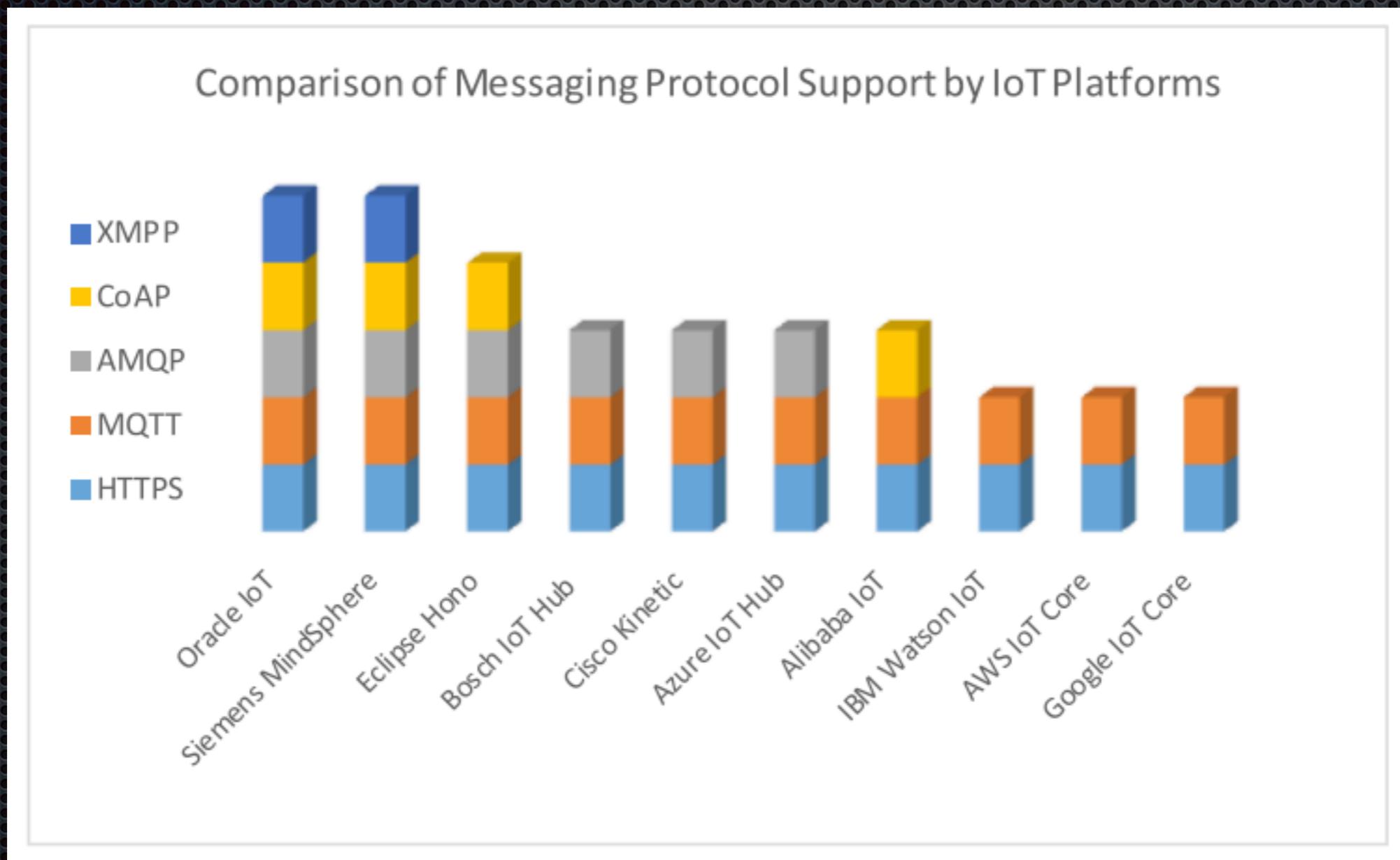
IoT Platform, <i>Year of First General Availability (GA)</i>	Protocol
Azure IoT Hub [166], 2014	HTTP(S), MQTT, MQTT over WebSocket, AMQP, AMQP over WebSocket <i>custom protocols transmit via a gateway</i>
Google IoT Core [167], 2018	HTTP(S), MQTT <i>custom protocols transmit via a gateway</i>
IBM Watson IoT [168] 2014	HTTP(S), MQTT
AWS IoT Core [169] 2015	HTTP(S), MQTT, MQTT over WebSocket, WebSocket
Alibaba IoT [170] 2015	HTTP(S), CoAP, MQTT, MQTT over WebSocket, WebSocket, <i>support network types: 3G, 4G, NB-IoT & LoRa</i>
Oracle IoT [171] 2016	HTTP(S), CoAP, MQTT, AMQP, XMPP, WebSocket
Siemens MindSphere [172] 2016	HTTP(S), CoAP, MQTT, AMQP, XMPP, <i>supports wide range of device protocols via field gateways (e.g. MindConnect) such as OPC UA, LoRaWAN, Modbus, 6LoWPAN, LwM2M</i>
Bosch IoT Hub [173] 2017	HTTP(S), MQTT, AMQP, LoRaWAN
Cisco Kinetic [174] 2017	HTTP(S), MQTT, AMQP, WebSocket <i>custom protocols transmit via a gateway (e.g. Cisco IoT Gateway)</i>
Eclipse Hono [175] 2018	HTTP(S), CoAP, MQTT, AMQP <i>uses AMQP 1.0 as primary messaging protocol</i> <i>custom protocols transmit via a gateway</i>

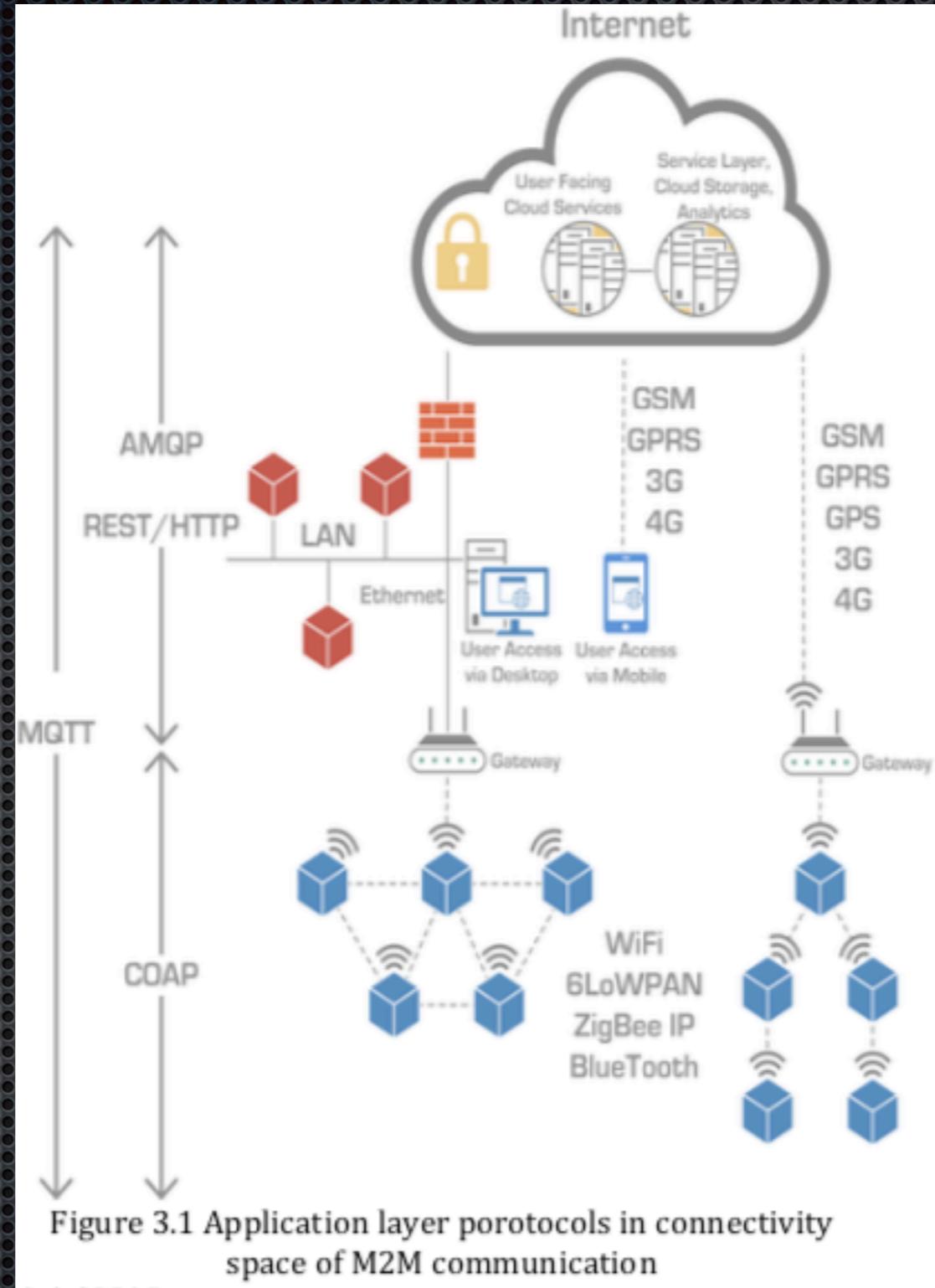
表：现有物联网
支持的协议/技术
列表

物联网平台支持的消息协议分布



物联网平台支持的消息协议比较





连接管理

- MQTT, COAP, HTTP/HTTP2
- 其他的协议?

< | 创建产品

基本信息

* 所属资源空间 booster_b8eb859d6182412f8f31a8772012e843 ⓘ

* 产品名称 BearPi_StreetLight

协议类型 MQTT CoAP HTTP/HTTP2 自定义 ⓘ

* 数据格式 二进制码流 ⓘ

* 厂商名称 BearPi

功能定义

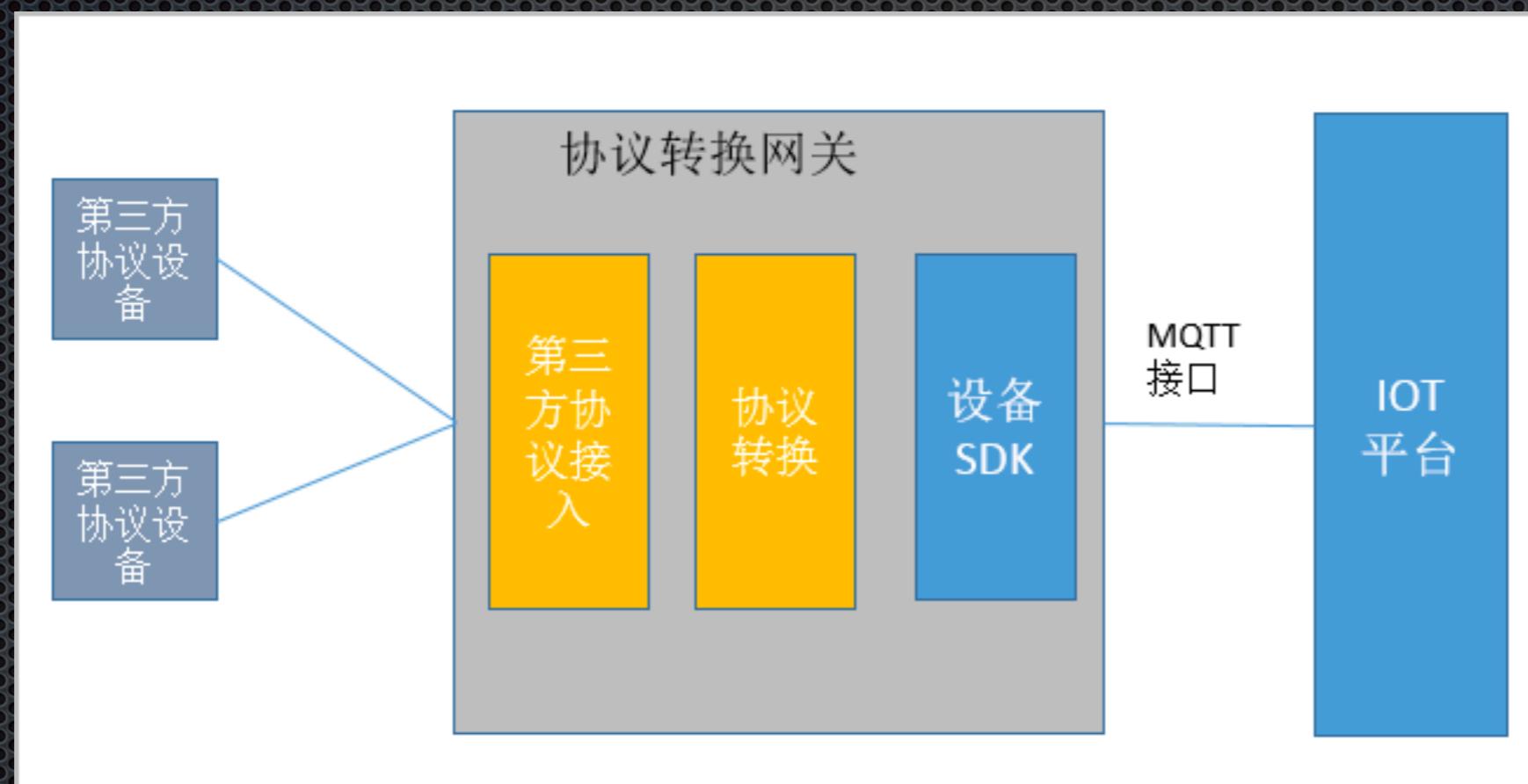
选择模型 使用模型定义设备功能

所属行业 智慧城市 ⓘ

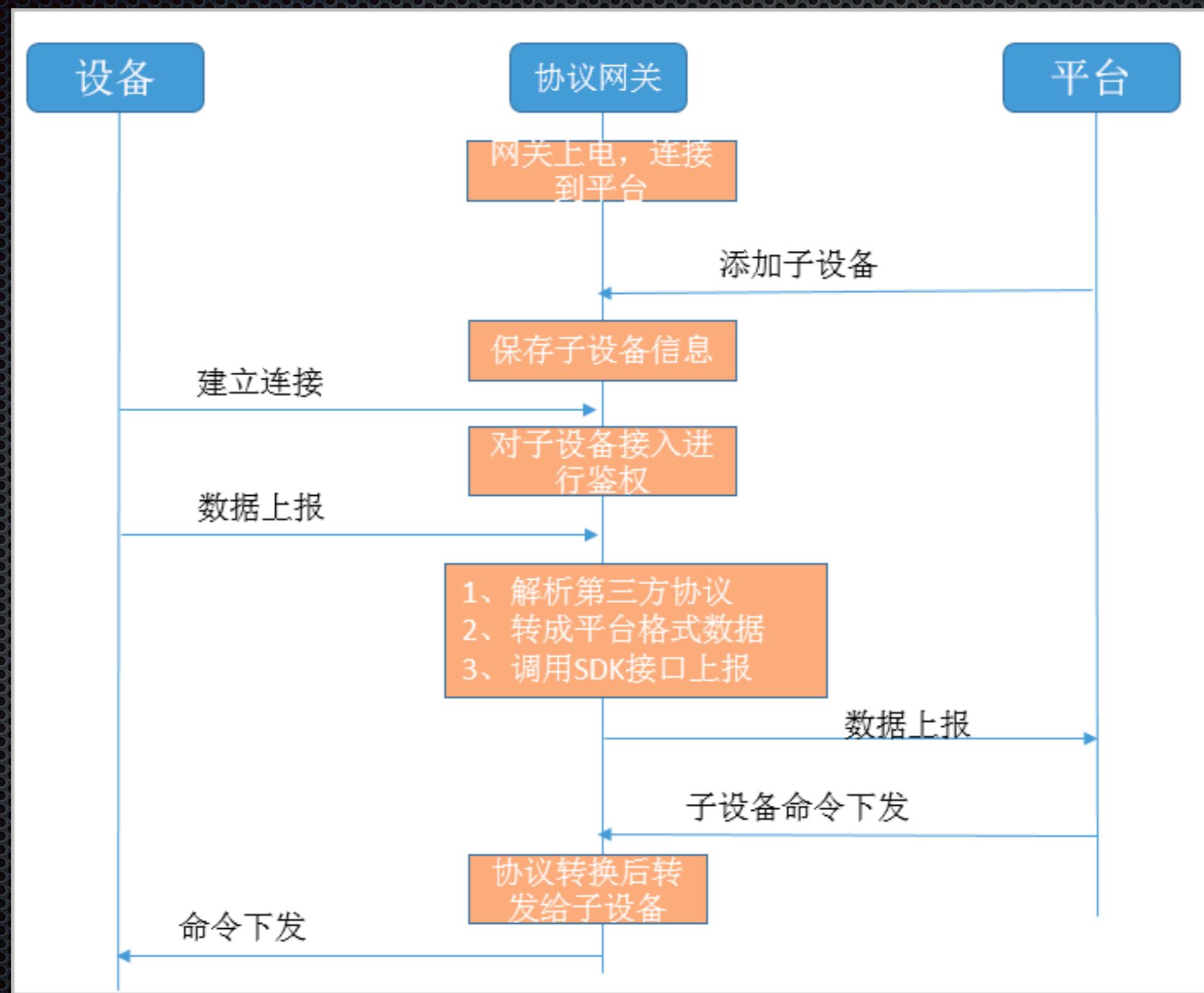
* 设备类型 StreetLight ⓘ

https://support.huaweicloud.com/bestpractice-iothub/iot_bp_0007.html

通过协议转换网关实现泛协议设备接入



https://support.huaweicloud.com/bestpractice-iothub/iot_bp_0009.html



参考文献

- Challenges and Opportunities in Edge Computing.pdf
- edge and fog computing for IoT A survey on current research activities future directions
- IDC: 中国边缘基础设施市场概览报告发布——边缘云发展正当时
- <https://zhuanlan.zhihu.com/p/55306124>
- <https://zhuanlan.zhihu.com/p/339727674>