OPEN INTERCONNECT CONSORTIUM℠ IoTivity

Cloud-Native Architecture and the Internet of Things

THE CHALLENGE OF SCALE MET BY CLOUD-NATIVE DESIGN

Attend any IT conference today and the recurring theme is: how to cut costs while supporting more services, in a more agile fashion? For several years, it was enough to migrate back-end servers running conventional applications into virtual machines rented from a provider like Amazon* or Microsoft*. But as workloads increased with the explosion of mobile devices, companies began to look deeper – how did Facebook* and Google* support so many users on their services in such a seemingly effortless way, scaling horizontally and reliably from one peak to the next?

Today's large cloud providers have responded to the question by releasing tools and describing techniques used to create applications designed from the beginning for massive scale. The meeting rooms of conferences are packed with attendees who are eager to learn these techniques, which are described as "**cloud-native**".

Cloud-native means, simply, that applications are designed to be deployed in the cloud from the beginning, easing horizontal scalability. An application that is architected to run on a single server then put in a VM and shipped off to a cloud service provider is not cloud-native and requires additional work to create significant scale.

Cloud-native applications are not monolithic with one central server providing hundreds of APIs. Cloud-native applications are composed of dozens of micro-services, each expressed as a RESTful API, whose provisioning code is in a software container, and whose lifetime is limited to the interaction with a single client.

This assumption of a lightweight, short lifetime is the key to architecting for scale and reliability. Open source projects like Node.js provide a platform for creating these services.

“Cloud-native applications are not monolithic with one central server providing hundreds of APIs”.

Projects like Kubernetes concern themselves with managing the life cycle of millions of software containers providing RESTful API services.

Cloud-native design has emerged as the only way to handle the load generated from millions of mobile devices.

OIC – CLOUD-NATIVE ARCHITECTURE FOR IoT

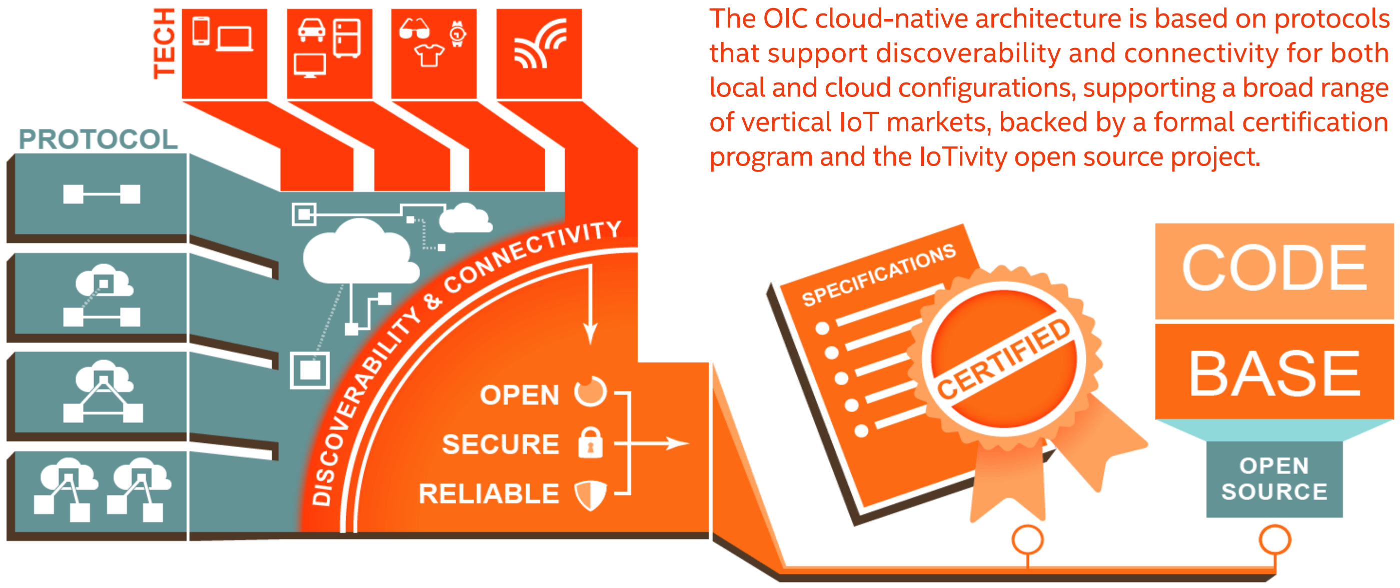
A company contemplating the Internet of Things may not think of themselves as having Facebook-like scale problems, but they will if their IoT projects actually succeed. A million devices are easily affordable when a CPU, memory and radio can fit on a fingertip and run on a battery for years. But without cloud-native techniques, a million devices can't be managed successfully.

This is why the Open Interconnect Consortium is the industry’s only cloud-native architecture for IoT documented in a formal specification that can be implemented virtually any way a developer chooses.

OIC 's cloud-native approach adapts cloud techniques for local use, enabling smooth integration between local and cloud (e.g., local-to-local and local-to-cloud). Product developers can decide how they want their products to take advantage of this flexibility. The cloud-native approach means that developers who start with a local-only approach don't have to start over if their products grow to embrace the cloud.

The specific techniques embraced and adaptedby OIC include:

- **RESTful API techniques that tolerate intermittent connections between devices and services**
- **CoAP (constrained application protocol) over TCP/IP that allows local devices to reach the cloud by sending information via TCP/IP to the cloud, while using the simpler UDP protocol locally.**
- **XMPP (extensible message passing protocol) originally developed for chat applications.**
- **JavaScript APIs implemented via Node.js in IoTivity to ease both local and cloud development.**
- **IPV6 and 6LoWPAN designed into the protocol from the beginning; provided there is a link to the internet, supporting devices can readily talk to the cloud.**



The OIC cloud-native architecture is based on protocols that support discoverability and connectivity for both local and cloud configurations, supporting a broad range of vertical IoT markets, backed by a formal certification program and the IoTivity open source project.

NOT EVERY DESIGN SCALES – ONLY SUCCESSFUL ONES

Some implementations of IoT do not require the level of scale mandating cloud-native design. Managing a few lights and speakers from a set top box or PC can use traditional techniques. But these first 'home management' scenarios are just the beginning of a complex and extensible inter connected world and ecosystem.

OIC's cloud-native design supports both local-only as well as cloud-connected scenarios with the same architecture. For example, an OIC network self-organizes if no cloud connection is available, using UDP instead of TCP/IP.

OIC does not mandate that all end devices communicate directly to the cloud. Gateways between sensors and cloud will make sense in many deployments. But the decision to deploy a gateway should be an optimization based on traffic, communication latency and cost.

"Only OIC's cloud-native design supports both local-only as well as cloud-connected scenarios with the same architecture".

Separating infrastructure choice from software architecture is the key lesson of cloud-native design from industry leaders. The wisdom of this separation is evident when applications like Google Maps, Waze* and Facebook* seem effortlessly available on our phones.

IoT projects that ignore these lessons can succeed – but only until they do, at which point the compromises in a design that is not cloud-native limits their scalability, reliability and business agility.

*Other names and brands may be claimed as the property of others.

Here is how alternative approaches to cloud-connectivity play out in a simple home use case:

Local-only:
PC or set top box is used to set up and manage local devices.

Cloud-only:
Smart phone is used to manage devices but only works if the phone and device each have an active internet connection even if the phone and device are on the same local subnet.

OIC:
Smart phone can manage the device in the home even if no external internet is available, just a common subnet. But the same phone application can manage the device remotely via the internet.