Migrating legacy M2M systems to the cloud

Chris Downey, Laird Technologies

The rapid pace of technological change means that, even before a system has matured, it often is replaced or enhanced with new technologies. Such is the case with machine-to-machine (M2M) applications and their migration to the cloud, or what sometimes is called the Internet of Things (IoT).

Just as with M2M, the definition of IoT is often in the eye of the beholder, but a few tenants hold true:
• The IoT is not just an extension of the internet as a whole but a new paradigm shift where machines, sensors, and controllers begin using common platforms which will further enhance their interoperability.
• Just as the Internet changed the way people communicate, the IoT will change the way machines communicate.
• The differences between legacy M2M communications protocols and IoT protocols can be stark. Designers must understand these differences as they migrate existing M2M systems into the cloud to partake in IoT.

The fundamental and obvious aspect of migrating a legacy application to the cloud is the change in perspective the designer must use. Everything is now global, so a system cannot be contained to a single network but must be viewed from the perspective of how all networks will communicate.

Addressing
A key issue is addressing. Unique addressing for each object is critical to being able

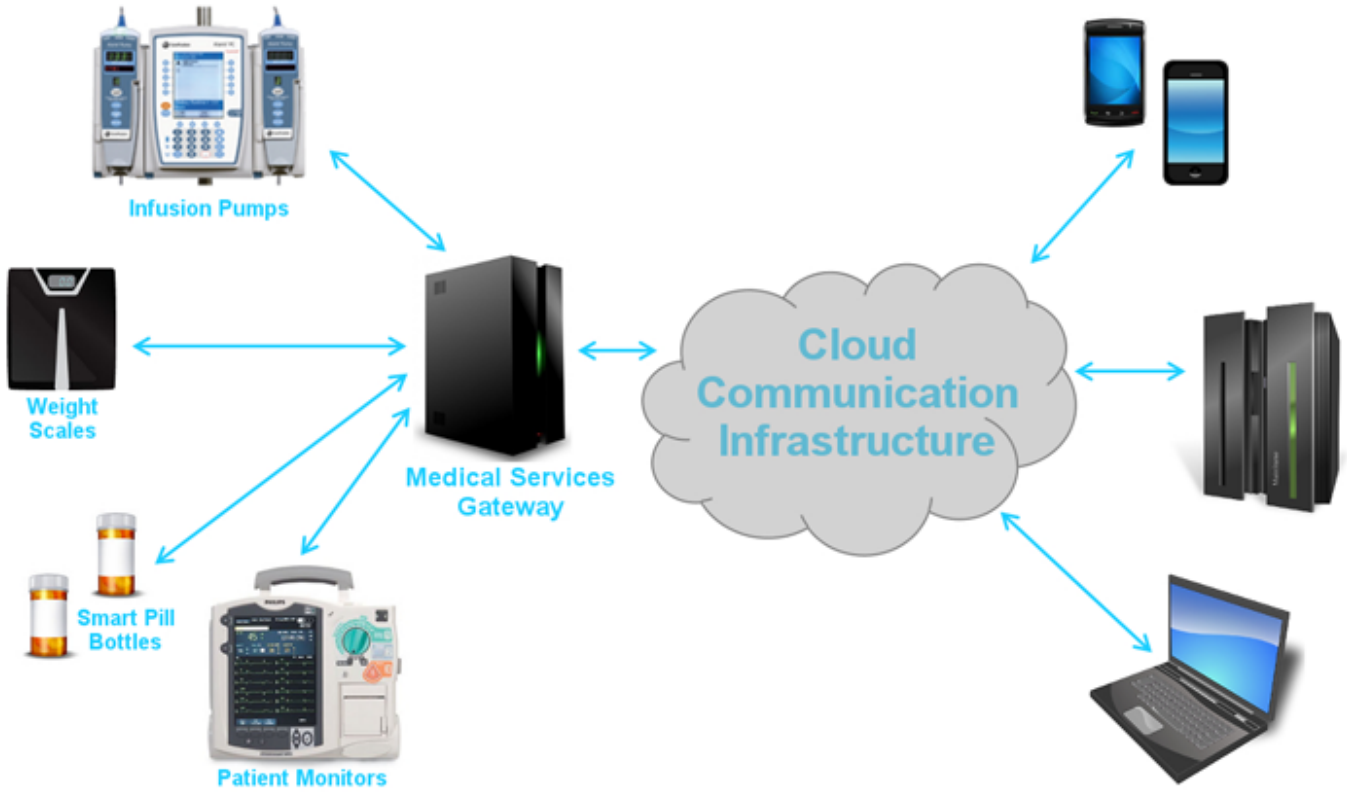
take advantage of a cloud-based offering. The Internet runs on the Internet Protocol and, in an ideal IoT system, each end node is identified by a unique IPv4 or IPv6 address. This enables direct communication between the cloud server and the end node. If a legacy application or system does not account for globally unique addresses, then a gateway device that translates end node addresses to IP addresses may be necessary. It is not necessary for every node to be identified by an IP address — unique addresses such as MAC addresses can still be used — but those devices will require a gateway which communicates via IP.

TCP and UDP

Another challenge when migrating from a legacy system to an IP is the use of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). Whether connecting directly to the cloud or using a gateway, eventually the packet will need to be transported via one of these two transport protocols and both may be different than the transport protocol you are currently using.

TCP, the most common protocol currently used on the web, will be used in many IoT applications even if it is not ideal for that application. That is because a variety of upper-layer protocols such as Representational State Transfer (REST) and Hyper-Text Transport Protocol (HTTP) ride on TCP. TCP provides for reliable, ordered delivery of information by creating a connection between two end points. With many Internet operations, as packets traverse the routed network, they become lost, out of order or delayed. TCP works to ensure the packets are reassembled in the correct order and that missing packets are retransmitted. The cost is latency — the entire message may need to be delayed while the packets are reassembled. This delay can be on the order of seconds as packets are retransmitted.

In contrast to TCP, UDP is a connectionless protocol that focuses on latency rather than reliability. With UDP, a user creates a datagram of a specific number of bytes, then that datagram is verified for reliability via a checksum field at the receiving end. If the datagram is corrupt or is not received, it is not retransmitted; retransmission must be performed by a higher-layer protocol. Likewise, because there is no sequence number with UDP, as there is with TCP, it is possible the datagrams could be received out of order. UDP often is used for broadcast and other unidirectional messages and, due to its low latency and lack of retries, for streaming services such as Voice over IP. Due to its lightweight nature, UDP requires fewer resources on the host system and may be the right choice for resource-constrained devices.



When migrating your M2M system to a cloud-based server architecture, you will need to consider the consequences of utilizing either a TCP- or UDP-based transport mechanism. If your device already has an IP stack, then this may not be a difficult decision, but if you have a non-IP device that you will need to utilize a gateway device, bearing in mind that such a device may have an impact on the reliability and the latency of the link. It is possible that you may want to use both methods, a UDP-based system for delivering data and a TCP-based system for reliable transport of critical large chunks of data, such as a firmware update.

Security

When migrating to a cloud-based server that operates as part of a global system, security is another consideration. It is highly recommended that all data be encrypted at either Layer 2 or Layer 3. If the end nodes of a legacy M2M system do not support end-to-end encryption, then encryption between a new gateway device and the server should be performed. The specifics of the encryption standard are best left to another article, but encryption should be used.

Authentication is another aspect of security and is just as important as encryption. It is likely that the authentication system for your M2M system is locally based and possibly restricted by physical access. It will be necessary to extend this to a server authentication which includes both device authentication (is the device allowed to connect to the server?) and user authentication (which users can access data about that device?).

Architecture

Finally, just as we started with a shift in perspective to thinking globally and not locally, we will end with the necessary shift in philosophy which should govern the design of a cloud-based server system. When designing for embedded systems,

there is a premium put on up-front architectural decisions. It is often difficult or impossible to upgrade systems once deployed in the field, so a solid system plan and test execution are critical to success.

With a cloud-based deployment, many of these issues are no longer as much of a concern. It is much simpler to deploy new code to the server that can be done on a daily basis or more often if necessary. In addition, the rapid rate of change in technology is very apparent in server systems with new mechanisms continuously being developed. Due to changing technology, having a complete design for a cloud-based system up front may be impossible. With this in mind, some leeway must be given to reducing the upfront architecture plans and focusing instead on developing code. Many software development systems such as Agile development and Lean Start-up advocate a more iterative approach to developing requirements and writing code. It is advisable to see how these philosophical approaches could be applied to your application.

Server-based systems have been around nearly as long as computers themselves, and migrating portions of that to a service-based model in cloud computing is a logical step which allows companies and users to focus on the most important pieces of the design. Once the data is migrated from the local network to a centralized server, a number of benefits from remote configuration, data visualization and collaboration with other cloud-based software all appear. These benefits will drive new applications and new value for customers, which make the cloud an attractive step for any M2M system.

Definitions sidebar
This article often uses the terms cloud and internet of things interchangeably, but that’s not quite accurate.

The term cloud should be referred to as consolidated offerings where some piece of the infrastructure is now being offered as a service. From the National Institute of Science and Technology (NIST) definition: “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. “Internet of Things" (IoT) is a more nebulous term which defies a standard definition; rather it includes the concepts of uniquely identifiable objects and how those objects use Internet Protocol to communicate. While there may be greater standards for interoperability in the future, the current notion is the idea that if we can virtualize these objects we can provide context and controls around them to make the whole greater than the sum of the parts.

Server is also used as a generic term for a computing system which consolidates information. If this is deployed in the cloud, some portion is offered as a service, but it could also be deployed on premise. Gateway devices are devices which merge two networks, one of which is an IP network, where the server resides.