A Common Language

Is a lack of standardization holding back machine-to-machine adoption?

M2M boils down to a two-step process—make a machine talk and listen to what it says. One step can’t work without the other, and to make them come together, a machine has to be able to speak the right language.

As you might expect, getting machines to speak the right language is a pretty big part of setting up an M2M system, and it’s also a major reason why more systems aren’t set up already. It’s not as simple as plugging in hardware and then using the machine data. Along the way, the data has to be formatted so it can leave the machine, travel over a network, and integrate with an application system.

Technology providers try to make that process as simple as possible. Their approach is to use standard data protocols so companies can focus on what to do with the data rather than how to get access to it. But standards haven’t completely solved the challenge of machine communication, and in some cases they have only made it more complicated.

“If it literally takes a network engineer to screw in a smart light bulb, pervasive computing is never going to work,” says Glen Allmendinger, president of Harbor Research Inc., www.harborresearch.com, San Francisco, Calif. “Many schemes and so-called standards for device connectivity already exist. But of course, all those solutions add up to one big problem. We don’t want many standards; we want one.”

This raises an interesting idea: could every machine speak the same language? Is there one universal dialect that would solve the dilemma of transporting and understanding machine data?

To answer these questions, we’ll look at the role of standards in M2M—what they do, where they’re needed, and who uses them. Then we’ll look at the standards that are used today and try to determine if machines really ever could have a common language.
Standards in M2M

There's a long list of standard protocols that are related to machine-to-machine communication, and the list is growing all the time. They all have their own nuances, but at the most basic level, the function of standards in M2M is to make it easier to transport machine data and for software systems to more efficiently interpret it.

By making it simpler to move data around, the theory goes, standards should make deployment easier and less expensive. The actual effect on the various stakeholders, though, isn't always so straightforward.

For end users, many of the machines they already own were programmed with legacy industrial-networking protocols that aren't designed to interoperate with today's communication networks and software systems. This means that to use data from those assets in an M2M system, an extra layer of software is required to translate the old protocols into a data language that can be integrated.

Meanwhile, standards also create complications for OEMs that are trying to network-enable their products. With a bevy of protocols their customers might use, manufacturers need to consider which ones their products will support and purchase communication hardware with embedded software that supports them.

"If you're going to allow application engineers to get information from a product, you want to make (milliamp). A physical connection is also required from the microprocessor to the communication hardware, and to make this connection, various standards are used for interface boards, ports, and connectors.

Next is the data-link layer, where machine data is readied for transfer over a network. Software protocols at this layer decide which data is sent and in what format by using a specific request-and-response message structure. Common standards at this layer include Ethernet as well as industrial-networking protocols such as Modbus, BACnet, DeviceNet, and OPC.

The next three stages that machine data encounters moving through an M2M system are the network, transport, and session layers, which are responsible for routing the data and making sure it gets to the right place. At the network layer, for example, TCP/IP (transmission control protocol) and UDP/IP (user datagram protocol) are two widely used Internet protocols used for establishing the data's route.

In applications that use wireless cellular networks, multiple competing standards exist at the network layer, including GSM (global system for mobile communications) and CDMA (code division multiple access) and their subsequent generations. In the case of local-area wireless systems, standards such as Wi-Fi and ZigBee govern data transfer.

The Protocol Stack

As machine data moves through an M2M system, it goes through several stages or layers, and each layer has its own set of standards.

First is the physical layer, where data has to travel from a sensor to a microprocessor or controller using circuitry standards such as 4-20 mA
STANDARDS IN M2M

Most commonly associated with data and programming languages, and they’re no different in M2M applications: XML (extensible markup language), HTML (hyper-text markup language), SMTP (simple mail transfer protocol), and SMS (short-message service), and Java, to name a few.

Within the application layer, several “M2M flavors” of various protocols have been developed, just as they have for other industries and applications. For instance, there are at least three M2M-related versions of XML, each with tags designed for performing common functions of machine-to-machine systems such as retrieving a sensor reading, turning on an actuator, or rebooting a device. The idea is that by standardizing the code programmers use for typical M2M functions, less time needs to be spent in application development when new systems are deployed.

One example is MIMOSA, www.mimosa.org, which is a series of standards for exchanging data between asset-management systems. Two others are M2XML, www.m2xml.org, which was developed by SensorLogic Inc., www.sensorlogic.com, Addison, Texas, and BitXml, www.bitxml.org, which was developed by Your Voice Spa, www.yourvoice.com, Milano, Italy.

To make a machine speak M2XML or BitXml, a programmer would install a driver for that protocol, typically in the communication hardware connected to the device. The complete specifications for the two protocols, both featuring a few dozen definitions, are available on their respective Websites.

What it All Means
In the last nine paragraphs, we’ve covered seven protocol layers and mentioned around two dozen standards that can come into play in typical M2M deployments. Obviously, an adopter probably wouldn’t use all of them in a single project, but it’s also just as unlikely it would use only one.

So how do they all tie together? For instance, let’s say a company needs to connect several devices on a wireless local-area network using the ZigBee protocol, and the devices are programmed to speak M2XML. How would the two standards interrelate?

“Saying you can speak M2XML over ZigBee is like saying you can speak English over a GSM or CDMA cellphone,” says Byron Appelt, lead developer for SensorLogic. “ZigBee carries the data and defines how it’s carried, while M2XML specifies data structures and transactions.”

Cellular Wireless Modems!

Multi-Tech wireless modems connect machine-to-machine (M2M) devices via EDGE, GPRS or CDMA networks. For seamless end-to-end connectivity solutions, you can count on Multi-Tech.

External Wireless Modems
- EDGE, GSM/GPRS or CDMA2000 1xRTT models
- 85-232, USB and Ethernet interfaces
- Desktop or panel mounting
- Carrier approved

Embedded Wireless Modems
- EDGE, GSM/GPRS or CDMA 2000 1xRTT models
- Universal socket design or rugged industrial chassis
- Easy integration
- Carrier approved

Multi-Tech has provided reliable modem connectivity for over three decades. Give us a call at 1-888-288-3309 or 763-785-3500 and let us put M2M wireless connectivity to work for you today!

Copyright © 2006 Multi-Tech Systems, Inc. All rights reserved.
carried, but the actual data itself is a different protocol—M2XML.

The whole idea of standardization is to have one way that everybody does something, yet dozens of standards are used in M2M, and some of them clearly overlap. While the bottom layers of the protocols are fairly well established, there is still very little standardization at the application level. In fact, there is a broad consensus among technology providers that most of the development work done in M2M is still largely customized for each adopter.

“I don’t think we’re even at that point where people have come up with standards at the application level that anyone is paying attention to yet,” says Steve Pazol, CEO of nPhase LLC, www.nphase.com, Chicago, Ill. “We have our own (protocols), and everybody else does too. Nobody has enough marketshare to force them on people, and that’s because the market’s still developing.”

With so many stages machine data has to go through, the possibility there could ever be a single standard for the entire protocol stack is quickly eliminated. But if there were a basic unit of device-data currency, as Allmendinger calls it, what would it look like and what would it replace?

At the application layer, this universal language would take the form of standard APIs (application programming interfaces), much like the tags developed by SensorLogic and Your Voice, although not necessarily using the XML protocol. As for the rest of the stack, there is one model that many believe is becoming a virtual standard for networking: TCP/IP.

“In my opinion, IP is the dial tone for the M2M industry,” says Benson Hougaard, vice president of marketing for Opto 22, www.opto22.com, Temecula, Calif. “If you make products that speak TCP/IP, you’re a long way toward making things work and communicate.”

The TCP/IP stack covers most of the stages machine data has to go through for companies to use it in an M2M system. But if machines aren’t speaking the language of the Internet already, then what are they speaking? The answer, at least among many of the machines that companies already own and use, are the industrial-networking protocols mentioned earlier: Modbus, BACnet, DeviceNet, OPC, and many others.

“I see certain networking protocols like Modbus slowly fading away,” Prowten says. “In the past, proprietary protocols were critical so you could talk to the device. But now there’s less emphasis because you can just transfer the information using TCP/IP.”

With Internet protocols acting as the backbone of machine-data transfer and one pervasive (albeit still unidentified) data language at the application layer, end-to-end M2M development could conceivably happen with only two primary standards.

Back in the real world, however, a more pragmatic approach to system development has taken hold—the common-object model. It involves taking heterogeneous data from machines and systems and translating it into software objects that can be incorporated into the TCP/IP stack. The end result is akin to an operating system for M2M: a standard set of tools for taking data from many different sources—regardless of what language they speak—and using it all to run a single system.

“We shouldn’t expect people to reinvent their backend infrastructure or reinstall everything on their machines,” Pazol says. “We need to be interoperable with existing standards versus trying to define our own standard and making them come to us. We need to go to the mountain.”

The pioneers of the common-object model in machine-to-machine begin with Tridium Inc., www.tridium.com, Richmond, Va., which has arguably the closest thing to an M2M operating system in its Niagara framework. Other leaders are still emerging with their own platform components, including Aeris.net, e3 Corp, nPhase, Numerex, and SensorLogic.

A consistent approach to making machines talk and listening to what they say is absolutely crucial to the growth and maturity of M2M technology. Ever since machines first attained the power of speech, companies have lined up to write their alphabets. The result is a list of standards so long it borders on the ridiculous. If a magic standard could exist, the industry is almost certainly a long way from being ready for it. In the mean time, one more standard probably isn’t going to make things easier, but be sure to keep reading M2M to find out who tries next.

<table>
<thead>
<tr>
<th>STANDARDS CATEGORIES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Machine Languages</td>
</tr>
<tr>
<td>Character Codes</td>
</tr>
<tr>
<td>Hardware Interfaces</td>
</tr>
<tr>
<td>Storage Media</td>
</tr>
<tr>
<td>Operating Systems</td>
</tr>
<tr>
<td>Communications &amp; Networking</td>
</tr>
<tr>
<td>Programming Languages</td>
</tr>
<tr>
<td>File Management Systems</td>
</tr>
<tr>
<td>Database Mgt. Systems</td>
</tr>
<tr>
<td>Text Systems</td>
</tr>
<tr>
<td>Graphics Systems</td>
</tr>
<tr>
<td>Internet</td>
</tr>
</tbody>
</table>

The same programming languages that are used to develop general software applications are also used to develop machine-to-machine applications. Most M2M apps are programmed using binary code, but XML is also used (read below). Java was originally conceived as a language for embedded systems, and Sun is making a push to get it in more M2M devices.

Give points to BitXml for ambition—its creator calls it “the ultimate M2M communication protocol.” It was released in June along with BitX, a set of platform tools for application development. The platform isn’t free but the protocol is, and v2.0 is expected soon. It’s unclear if M2M is ready for a single development standard, but BitXml’s approach clearly addresses the right issues.

It may not be as polished or comprehensive as BitXml, but M2MXML is also not as verbose. SensorLogic designed the platform with cellular networks in mind, and as a result it uses as few characters as possible to execute typical M2M functions. So far, most of the users of M2MXML are SensorLogic customers, representing around 1,000 total devices.

Of the three players in this group, MIMOSA and OPC actually develop standards. The other, Open O&M, is an initiative founded by the other two to coordinate each other’s efforts. OPC standards cover data acquisition and transport; MIMOSA is for exchanging data between asset-management systems. In May, MIMOSA was mandated by the U.S. Dept. of Defense.

Tridium’s Niagara Framework isn’t an open standard, but it’s the closest thing M2M has to a standard operating system. It’s essentially a repository of interfaces, application elements, and other tools. Developers use the platform to take machine data from heterogeneous devices and systems and convert it to software objects that are understood by an M2M application.

Industrial and building-control standards traditionally go hand in hand, but as the space evolves, so do the standards for building-automation systems. LonWorks is a widely adopted binary protocol used primarily for building control, oBIX serves the same purpose but by using XML to link the enterprise with mechanical and electrical systems.

Web services are standard software components designed for integrating applications with other applications. In M2M, they’re used to integrate machine data with enterprise applications. For instance, an ERP system isn’t designed to accept sensor data, but it can recognize Web services, so middleware companies use those standard interfaces as the go-between.

Machine data often turns up on the Web in the form of a dashboard for monitoring an asset’s performance. Developers use AJAX so the page doesn’t have to reload every time the data changes, and they use RSS to combine data feeds from multiple sources. Meanwhile, SMS is a common method of using cellular networks for sending alerts from remote assets.
A COMMON LANGUAGE: PROTOCOL PROFILES

Group: Internet Protocol Suite
Standards: SMTP, SNMP, TCP/IP, UDP/IP
URL: n/a

Much of the grunt work in M2M is focused on taking machines that were programmed to speak proprietary networking protocols and enabling them to communicate using Internet protocols. Many believe the legacy protocols will eventually give way completely. TCP/IP is more widely referenced, but insiders say in M2M, UDP/IP is more widely used.

Group: Standard Access Method
Standards: Ethernet
URL: n/a

As a standard, Ethernet covers a wide range of networking technologies and multiple layers within the protocol stack. For machine data, Ethernet refers primarily to wiring and signaling standards at the physical layer, and it’s becoming widely used in automation systems. It’s also the primary transport mechanism for wired M2M, often combined with wireless networks.

Group: Short-Range Wireless
Standards: Bluetooth, Wi-Fi, WiMax, Z-Wave, ZigBee
URL: www.z-wavealliance.org; www.zigbee.org

We chose five short-range wireless standards for this group, but the list could have easily been twice as long. Z-Wave and ZigBee are competing protocols for sensor networks, with Z-Wave mainly focused on home automation. WiMax is seen by some as Wi-Fi’s eventual replacement, and Bluetooth is the current standard for personal-area networks.

Group: Wide-Area Wireless
Standards: CDMA, EDGE, EV-DO, iDEN, GPRS, GSM
URL: www.ctia.org

The cellular generations continue to advance and largely to the benefit of M2M adopters. For the last two years, carriers have rolled out their data networks, EV-DO (CDMA) and EDGE (GSM), giving them incentive to drive more machine data traffic onto their networks. The results are more options and more competitive rates as carriers climb onboard with M2M.

Group: Certification Standards
Standards: CDG, GCF, PTCRB, R&TTE
URL: www.cdg.org; www.ptcrb.org

To certify a device for use on a cellular network in North America or Europe, developers have to pass a series of tests with one of three certification standards organizations. CDG determines requirements for CDMA certification, PTCRB controls GSM/GPRS certification, and R&TTE handles compliance issues in the European Union.

Group: Modem Instructions
Standards: GSM/CDMA Protocol Stacks
URL: www.wavecom.com/OpenAT

Machines know which data to retrieve and send based on instructions from software that usually runs on the communications hardware. For wireless cellular modules, those instructions are called AT commands. GSM modules providers largely use their own proprietary stacks to differentiate their products; Qualcomm is the main software provider for CDMA modules.

Group: Industrial Networking
Standards: BACnet, DeviceNet, Modbus, Profinet
URL: www.bacnet.org; www.modbus.org

Most of the machines deployed today use protocols that were written before the Internet went mainstream. Naturally, they weren’t designed to speak Internet protocols, which have become the primary standards for enterprise-system integration. Still, legacy protocols have a strong hold in industrial automation and until recently, machines couldn’t talk without them.

Group: Industrial Wireless
Standards: ISA SP100
URL: www.isa.org

The ISA-SP100 committee says it will establish standards for technologies and applications that happen to be at the very heart of M2M, including some that are already represented by other existing standards. The ISA (Instrumentation, Systems, and Automation Society) has a strong hold in automation and its work could strongly impact M2M for better or worse.