Proceedings of the Embedded Systems Workshop

Cambridge, Massachusetts, USA, March 29–31, 1999

Bringing the Internet to All Electronic Devices

Michael Howard and Christopher S. Sontag

emWare Inc.

Bringing the Internet to All Electronic Devices

Michael Howard
Chief Architect, emWare Inc.
Christopher S. Sontag
CTO and co-founder, emWare Inc.

Abstract

In order to develop appropriate solutions for embedded device networking, we must understand the benefits offered to the end user of the device as well as the costs involved with delivering a solution. As proponents of networking technology, it is tempting to overestimate the perceived value of connectivity, while at the same time overlook the hidden costs in implementation. It is important to remember that companies adopting technology for embedded device networking are manufacturers who tend to be very conservative due to narrow margins and fierce competition. A prerequisite to wide adoption of this networking technology is a convincing case for a strong return on investment. Therefor, all optimism must temporarily be put aside for a critical cost/benefits analysis that will provide criteria for judging the suitability of proposed solutions.

Since there are so many embedded applications, it is not surprising that quite a few applications are 'no-brainers' which require little deliberation as to value or implementation method. Administration of networking hardware, for example, is increasingly accomplished through Internet-based device interfaces.

Internet standards and associated technologies provide a remarkable set of opportunities for enhancing the value of existing and new embedded products. These Internet technologies have been available to millions of large, 32- and 64-bit systems for some time. The key challenge we have undertaken is to establish a distributed device-networking platform that provides appropriate solutions for all embedded devices. Our goal is to make connectivity practical even for systems that do not have convenient networking already available and are severely constrained by economic pressures.

1.0 The Problem

Networking embedded devices is a tremendous challenge because the size of devices, the kind of processors, the type of network, and the information to be extracted from devices vary greatly. The device itself can't bear the majority of the burden of being networked. The device can't bear the majority of the overhead to provide security and it can't bear the burden of a too-high price point. And, in some cases, there may not be value in devoting resources to enable a device to do much more that it already does – due to the simple economics of low cost electronic devices. Technology must adapt to the reality of low-cost devices, not the other way around. Current solutions are too specific to either a certain programming language, physical network wire, transport communications protocol, or operating system. For example, http has become the solution for the Internet because it's generic. We can learn from this model. Many aspects of the problems are the same—the variety of interfaces, the range of processor capabilities, the need for real-time state communication and connection-based control, and the batch data collection and datagram dissemination.

In order to develop appropriate solutions for embedded device networking, we must understand the benefits offered to the end user of the device and the costs involved with delivering a solution. As proponents of networking technology, it is tempting to overestimate the perceived value of connectivity, while at the same time overlook the hidden costs in implementation. It is important to remember that companies adopting technology for embedded device networking are manufacturers who tend to be very conservative due to narrow margins and fierce competition.
A prerequisite to wide adoption of this networking technology is a convincing case for a strong return on investment. Therefore, all optimism must temporarily be put aside for a critical cost/benefits analysis that will provide criteria for judging the suitability of proposed solutions.

Since there are so many embedded applications, it is not surprising that quite a few applications are 'no-brainers' which require little deliberation as to value or implementation method. Administration of networking hardware, for example, is increasingly accomplished through HTTP-based device interfaces. Our goal is to make connectivity practical even for systems that do not have convenient networking already available and are severely constrained by economic pressures.

The cost of implementation results not only from the hardware and software used in communication, but also from:

- *The complexity of a system that must be understood by the applications engineer.* In situations where no device networking is currently in place, a great deal of education must happen in order to integrate communications into an application. The impact on the application in terms for timing, power consumption, noise, and memory consumption must be understood and dealt with.

- *Education of the end user in terms of marketing, documentation and customer support.* The person who stands to benefit from the networking of embedded devices is very likely unaware of the advantages a networked system has over an isolated one. The value added to an application needs to be communicated in order to influence purchasing decisions. Products that might have been closed to configuration and monitoring must now be explained in greater detail in order to enable more intelligent use. Customer support must effectively deal with a new set of problems in order to insure that this new technology does not prove more trouble than it is worth. The cost to a user is not simply based on dollars, but also on the added time and attention that may be required.

- *Tools required for implementation and production.* A set of non-recurring costs for implementation of networking includes all the development tools and changes to a manufacturing process required by the technology. Packaging and government required process changes may be considerable.

- *Embedded device hardware costs for additional resources.* The addition of communications capability will likely require the addition of hardware, including additional CPU power, memory, communications hardware, connectors, and power supply output.

- *Continued support of interfaces that rely on components and are changing over time.* If an industry standard Web browser is used to implement a device user interface, then the manufacturer of the device is burdened with the need to continually insure that changes to the browsers supported do not break the interface delivered to the users of their product. In addition, the browsers must support components like a Java VM. If the support for a particular language feature is altered in any way, the device interface may need to be updated. There seems to be an inherent incompatibility between the rapid evolution of software components and the life of an embedded system. The manufacturer must predict the cost of any such burden. This cost is particularly troublesome to estimate.

- *Additional development and test time to integrate communications functions.* As development cycles are shortened due to market pressures, the adoption of technology that might, at least temporarily, lengthen the cycles for a particular company may cost that company in terms of missed marketing opportunities.

The product developer who must predict total implementation costs evaluates these costs in terms of recurring costs versus non-recurring costs. The manufacturer of high volume products must pay careful attention to the cost of materials and labor, spreading the engineering costs out until the engineer cost per product is insignificant. The products that are created for smaller markets must recover a larger portion of these costs per unit. In order to facilitate rapid adoption of device networking technologies, we need a flexible solution that minimizes engineering costs or material costs.

The risk of adopting communications technology, which later proves unpopular and results in users demanding change to the alternate technology, is also a consideration. The NRE costs for switching a product to a standard will be weighed against the benefits of how early communications capabilities will be available.

After accounting for all the costs associated with networking an embedded device, we can move on to tackling issues such as the appropriateness of single transport, processor, networking protocol, or human or machine interface. The solution lies in creating an economically viable solution that provides a single mechanism to interface with devices and accommodate the methods of transport—any device, running across any transport, to communicate with any application or interface.

2.0 The Solution: Internet-based Communications System to Interact with Devices

Although networking devices for the purpose of distributed control and data collection is not a recent phenomenon, the popularity of the Internet has driven massive development of interface and other networking software. The resulting tools and general awareness of networking has made widespread device networking much more practical than it has been in the past.

The benefits offered by networking embedded devices fall into the following four general categories:

- *Low cost user interface.* A physical keypad and LCD display costs more as features and capabilities are added. In some cases, additional product features may make a device interface difficult to understand due to a proliferation of buttons and indicators. A browser-based interface, however, can be layered into simple control panels that relate to the current state of the device, and can be configured for an individual's needs.

- *Links to information available on the Internet.* Product documentation and upgrade information may be integrated with the user interface. This can be much more helpful than a 'click here for the manual' system. The interface application can query the device and offer meaningful help and upgrade information. A historical database may be accessed to provide expert system features. An example would be a car diagnostic interface that would be greatly enhanced by information from other vehicles of the same type. Another benefit of integrating applications with available Internet resources is that new product features may be created. For example, a sprinkler controller enhanced with weather prediction and historical data will allow conservation based on weather predictions.

- *Value of information on device not previously available.* Products that use embedded controllers often have some information that could be useful. As trivial as it may sound, things like the temperature of a furnace may indicate the need for filter replacement. Similarly, the duty cycle and outside temperature of a refrigeration unit may indicate a cleaning is needed. These are not exciting product features that drive consumers to replace old equipment, but they may reduce the cost of ownership and improve the product. The remote availability of performance and status information will help in diagnostic and configuration support. Information in the device may be useful in terms of providing usage information to marketing and advertising groups. A network of vending machines, for example, will provide feedback regarding the effectiveness of an advertising campaign. The same usage information may be needed to schedule restocking of product that has been vended. Finally, in systems whose purpose is the collection of data, remote connections reduce the price over manual collection efforts. The need of the power metering industry best illustrates this point.

- *Value of remote device control.* The convenience of having remote access and monitoring of various systems can be an important product feature. In some situations, a product may not have an interface appropriate for interaction with a microcontroller, so the addition of remote access allows manipulation where none was previously possible. The embedded firmware engineer is sometimes faced with the responsibility to implement or choose a configuration that is the best setting for most users of a particular product. Once chosen, this setting becomes an immutable product characteristic. Access to configuration values provides a mechanism to provide product that more closely fits the user's needs. A device control panel may be viewed as such a decision. If an owner of a particular device could toggle between a simplified or sophisticated interface, the *world would be a better place*.

The points mentioned above illustrate an important point—the benefit derived from the communication capability may effect the parties involved with maintenance and support of a product, not the end user. This becomes important when assessing the value these new features have in influencing purchase decisions.

2.1 Increasing the value of Communication capabilities in a device

Networking technology has some benefits that are derived from the wide availability of other networked devices. Although these would not currently tip the cost/benefits scale, these factors will gain in importance as more devices become available through a network.

- Use a common client interface for devices from various vendors, or for various models of a device from a single vendor. When a particular interface for a device provides access, but does not expose features or controls that are unique to the device, then the development of a mechanism for using a common client interface will save product development costs. This seemingly obvious point, which has driven development of device driver architectures, database object models, and TCP/IP application services, seems to have had very little impact on device control architectures.

- The machine interface of a device, in the form of a communications API, allows the device to be incorporated into a larger control system which may be composed of many other embedded devices. The ability for a particular device to participate in such a system may be a product feature that increases the perceived value of the product in the eyes of an end user. The behavior of multiple systems may be coordinated in new ways. For example, having a car stereo fade to a lower volume when a cellular phone call comes or load shedding during peak power demand times.

- The value of communications to an embedded device can be real even with only one device. A Weiser Lock (see figure 1) can be of value without any other devices connected to the same sub-network (i.e. a remote cabin can be programmed with temporary access for weekends renters.) There should not be a high cost to implement a small network (or a single device network) as a larger network. At the same time, if the Weiser Lock is added to a security system device on the same network, the security system can now show you that the door is closed and locked.

After evaluating the cost of implementation and value of access, it may be determined that no suitable connection technology exists for a particular application. It is our task to evaluate the world of embedded applications in terms of cost/benefits and create a strategy for technology development that will facilitate adoption of embedded networking technology.

The value of the communication function will dictate the set of available communications hardware and software solutions. It is unreasonable to expect a solution that costs more than the perceived value of the connection to be adopted.

3.0 The Gateway Approach

A gateway system provides an embedded architecture that addresses the issues discussed above and provides networking capability to even the smallest of embedded controllers. By establishing a gateway that acts as an intermediary “broker” between lightweight device networks (RS232, RS485, Modem, IR, RF) and heavyweight networks, including intranets and the Internet, you off-load the device and achieve the desired communication. The gateway can reside on a PC, single board computer, handheld, or a device with sufficient resources (32-bit microprocessor). The gateway, having more resources, can augment a device and enable more information and data to be exchanged.

A gateway provides device access and management services to nodes on the Internet or an intranet. This gateway is a TCP server that has the following set of responsibilities:

- *Device Firewall*. The devices rely on a gateway process to provide authentication and encryption where appropriate. In addition, the gateway can augment the security on the device by enforcing the most current security policies in this rapidly changing, Internet-connected world. Establishing security to protect the embedded device from unwanted users is necessary, but it is also critical to protect the network from undesired behavior of a device. If a device has unrestricted access to services on a network, then the device may pose a security risk. Whether through malice or oversight, a device could compromise the integrity of a secured subnet. The device gateway is charged with the task of securing access to a device and enforcing a usage policy on the embedded application.

- *Protocol Translation*. The gateway is responsible for adapting to the variety of device connectivity hardware and software solutions and delivering a uniform TCP/IP based access service. Thus, devices with existing communications software and hardware may be made available to network-based clients.

- *Device Watchdog*. A service of the gateway is monitoring device status. Whether the devices are to remain connected at all times and the gateway reports failure of the ancillary network or device, or the device connection is required to periodically contact the gateway process, the gateway reports to a designated client when a device becomes unavailable.

- *Event Handling*. A particular event from a device may require a launch of an application, or dispatch of a pager or e-mail message. In this situation, there is no connection to a "client" waiting for a message.

- *Device Services*. Instead of viewing the network as a client of an embedded application, it may be desirable to offer a set of network services to an embedded application. The gateway acts as a provider of services to embedded application. These services include database storage and retrieval, communications to network applications and other devices, device application extensions, and firmware updates. Some applications, such as automobiles and remote monitoring equipment, do not lend themselves to permanent hardwired connections. These mobile and remote applications may have persistent network storage in the form of a database, which may store commands, state, and history information that is always available to client applications. By de-coupling the device from a client that is interrogating the device state with a state database, the reduction in redundant traffic to the device is reduced and a point of interface for clients wishing to connect to a device may be down. A stock interface may respond and deliver information regarding the unavailability of the device. The batch- oriented paradigm, where the user essentially says "change the following settings next time it checks in" is coupled with a watchdog function and results in more efficient use of machine and human client's time. While this may be implemented in a standard email system, some extensions would be desirable, including the ability to signal the device, when it calls in, to remain connected while a client, that has previously requested a direct connection, is alerted.

Five aspects of a gateway-based system to consider include the following points.

3.1 Any Processor

There are more than 1,000 “flavors” of microcontrollers currently on the market. Each one is designed to cut costs and service a certain set of functions. There is no single processor or processor family that can service all embedded system requirements. In many cases, assumptions are being made in the embedded market that higher-end processors are required for any form of communications external to the embedded device – but the major logic flaw is that this is OK due to the ever-lowering cost in higher-end processors due to Moore’s Law.

Moore’s Law is being misapplied. We can’t assume that 32- and 64-bit processors will drop so low in price—to pennies—and still deliver incredible performance. Even if cost is not an issue, there are still the issues of power consumption, package size, thermal dissipation, and other “embedded” requirements that are difficult, if not impossible, for larger processors to satisfy. At the same time, Moore’s law also applies to all microcontrollers. These smaller MCUs are gaining additional capabilities while also costing less. Every device and application has different needs, therefore, a large array of processor choices with different performance, power requirements and capabilities is critical.

3.2 Any Capillary Network

Lightweight, capillary networks using transports such as RS232, dial-up, 485, RF, IR and CAN must all be supported to provide adequate options for networking 8- and 16-bit devices. Just like embedded processors, there is no single physical network transport that is appropriate for all environments. Whether it is cost, communications robustness, security, or a myriad of other reasons, solutions for networking embedded devices must support a wide variety of networking types that offer a range of capabilities. In addition, networks for 8- and 16-bit-based devices need to be able to operate in low-power, transient environments where devices may or may not have continuous access to a network. The method by which these lightweight networks communicate with larger, wide-area networks, including the Internet, must be efficient and timely. Unlike Ethernet and TCP/IP, which are proving to be an effective single-communications method for larger computer systems, there is no single lightweight network that is appropriate for smaller electronic devices.

3.3 Flexibility

Flexibility - EMIT software architecture is flexible and scalable. EMIT software can be configured for a wide variety of applications, including existing devices and proprietary networks. The user interface can operate on a remote Web browser, a directly connected laptop, or even a handheld PDA. Where a user interface is not necessary, it can connect directly to an application or database. By utilizing a gateway approach, implementers can also have flexibility on where interface descriptions are stored. With a gateway, the interface can be fully described and stored on the gateway, eliminating any resource requirements for storing the interface on the device. Or, the gateway can be used to augment partial interface description such as a tagging method. emWare’s® patent-pending Microtags are stored on the device yet reference pre-built interface components—such as emWare’s emObjects—that can be called from the device but stored on the gateway.

3.4 Any Interface

There are countless methods to interface with devices including Windows, the Web browser and the phone. There is no such thing as the "ubiquitous interface" for smaller electronic devices. For certain types of applications, a simple, text-based HTML interface may be sufficient, however, in other types of applications it may be overkill or not sophisticated enough. To provide rich, device-networking solutions, human and machine interface options need to be provided that scale from DTMF—phone key—to complex enterprise databases and resource planning applications—such as SAP R-3. In addition, many of these "interfaces" may be used with these devices at the same time.

For example, emWare recently demonstrated a series of networked vending machines at an SAP worldwide conference. This demonstration, built on emWare's architecture, was a collaborative effort by SAP, IBM, Sybase, 3Com, and Micron. The vending machines could be accessed by a Web browser. In addition, the vending machine interfaced concurrently with a number of other interfaces and applications. IBM demonstrated voice recognition technology being used to "drop a can" (from the vending machine) through the emWare connection. Likewise, Sybase showed database integration for inventory management, and SAP showed enterprise application integration for scheduling service routes to the vending machines. 3Com's Palm Pilot was used through an IR interface to directly connect with the vending machine and get and set configuration parameters. In addition, Micron's near-field ID badges were used to demonstrate contactless, microcash transaction with the vending machines. They all leveraged emWare to allow their unique interfaces to interact with the vending machine device at the same time.

3.5 The Object Model

An object model for lightweight devices is required to accommodate the diversity of embedded devices, the different transport systems and collaboration requirements, creating the building blocks for making the integration of multiple object environments (CORBA, JINI, and DCOM) possible. This will allow for even greater sharing of information and control.

The end result for embedded device communication should be an object representing the capabilities of the device. Everything that the application engineer wants to make public is exported from the embedded application to a device object server and then through a Gateway to applications on the Internet or an intranet.

In many applications it is desirable to treat embedded devices as abstract data types. By encapsulating functional behaviors of a particular device type, and standardizing around these encapsulations, we create an environment that will foster the development of network-client applications. These applications use an object model and definition of a particular application function to de-couple the client interface from the manufacturer's device. The gateway provides access to the device through an object service, and isolates the client from network and device-specific considerations. A home management software package, which could run on a PDA or browser, may access a heating control unit based on the object definition of an HVAC. The gateway provides the necessary object interface to an HVAC and isolates the client from the specific network and device characteristics.

Several choices exist for representing embedded devices. These choices include open standards like CORBA, and proprietary standards including Sun's Jini, Microsoft's DCOM, and one currently in development at emWare that is targeted for smaller embedded devices. All of these systems require that industry-standard object definitions be created in order to abstract the functionality of a particular device. Like all of emWare's architecture, the object-oriented implementation does not impose large resource requirements on the device. However, emWare's object definition can also be linked with Jini, CORBA, or DCOM to provide enterprise-wide, object applications.

4.0 emWare's EMIT®

For example, emWare has developed Embedded Micro Interface Technology (EMIT) that provides developers and manufacturers with a cost-effective and flexible means to easily implement Internet capabilities into their 8- and 16-bit microcontroller-based devices. The three main components—emMicro, emGateway, and the interface application—provide a complete and flexible device-to-user solution.

emMicro is an extremely thin, device object server that requires minimal resources at the device by taking up as little as 1 Kb of space at the device. The exception-handling mechanisms for the object provides a way to signal alarms, request services, and perform other device-initiated actions. With the addition of emMicro in an embedded device, the device is now instructed to expose any desired variable, event, function, or document from the device to any application or interface via a network object methodology. emWare sees distributed network objects as a profound refinement in communications technology for embedded devices.

emGateway compliments emMicro by serving as a bridge between the lightweight device network and the Internet or intranet. emGateway also provides the additional services for multiple-device management and Internet communication with a standard Web browser, or other interface applications or databases. The Web browser can use pre-built interface components (emObjects) to display and interact with the device in an intuitive, straightforward manner without requiring large amounts of resources on the device. emGateway can also be distributed over the network, allowing for many communications and resource-shedding options. Other situations will require that emGateway be combined with a Web browser (on a laptop, for example) to directly connect to the device. The flexibility of EMIT's architecture allows the components to be distributed and installed where they make the most sense, depending on the situation.

emWare's goal is to identify the parts of the embedded application to export, then create a network object that represents this interface to the device. Given that, we must question the utility of implementing this on top of a large communications stack that is made general for the purpose of simultaneously supporting other communication paradigms. More traditional and general implementations result in higher costs and device requirements.

5.0 The Next Steps

The next frontier is to move beyond simply communicating with devices and create an environment that enables devices to be treated interchangeably. For example, this step requires an extensible architecture that enables an interface to be created for any device, such as an HVAC thermostat, without the environment needing to know the chip, manufacturer or vendor of the device.

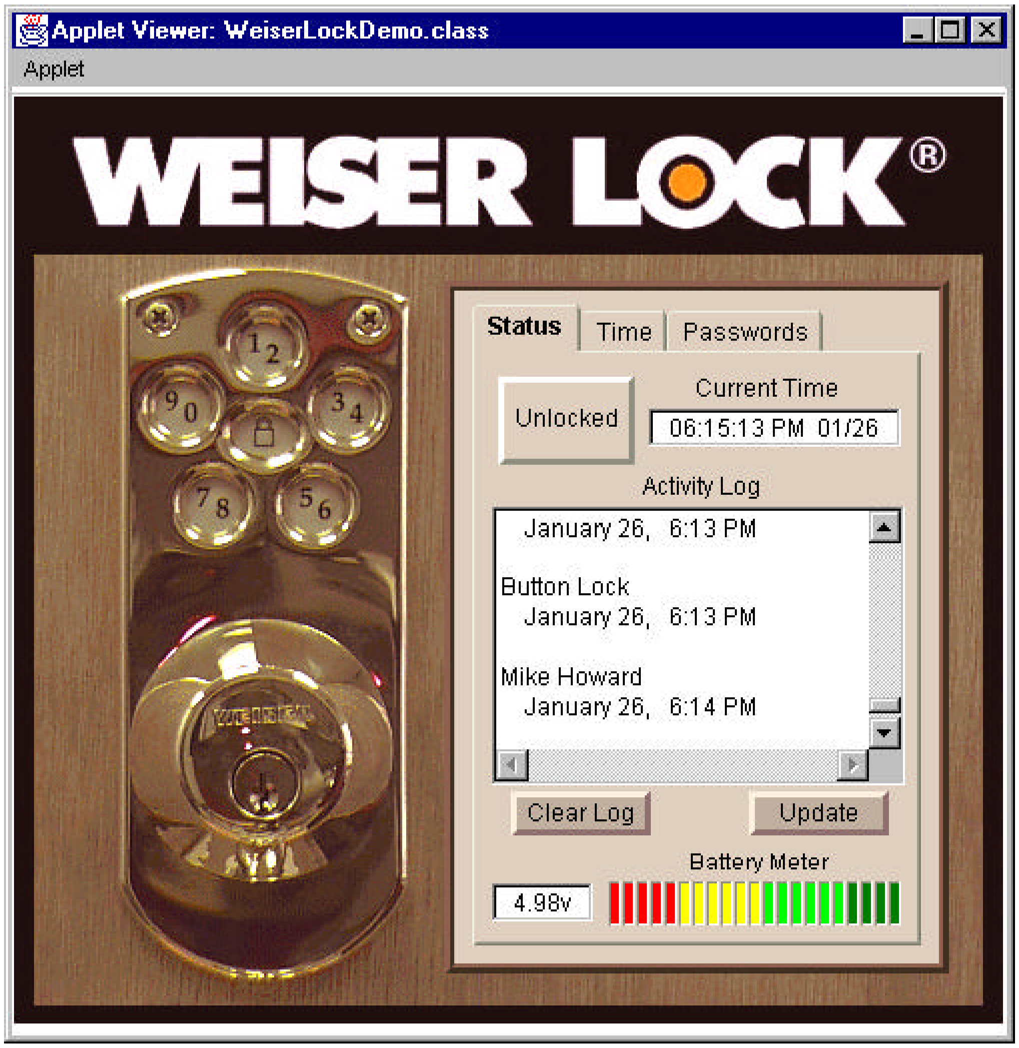
5.1 A Collaborative Effort

Delivering on the promise of device network technology and all that it has to offer the world requires a tremendous collaborative effort. The cooperative efforts of microcontroller, software, hardware, database, enterprise systems, and user interface companies is required to bring all the pieces together in a cohesive working solution. For this reason, emWare has formed the “Embed The Internet” (ETI) Alliance to provide a forum for collaborative efforts between some of the largest companies in the world, including Hitachi, Philips, Analog Devices, Microchip, Atmel, Siemens, Mitsubishi, National Semiconductor, Motorola, SAP, Sybase, Phytec, Centura, TASKING and Pervasive. emWare has also started a working group to explore standards opportunities in the 8 and 16-bit microcontroller-networking market.

6.0 Summary

Bringing the Internet to all electronic devices, especially the billions of 8- and 16-bit devices, is a seemingly daunting challenge. The size of devices, the kind of processors, the type of transport and the information or data to be extracted from devices varies greatly. Solutions to date have been too specific and addressed just one of the variables, such as language, wire, transport or OS. The gateway system with its modular architecture, distributed approach and complete integration with microcontrollers, software, hardware, databases, user interfaces, and enterprise systems provides the framework for virtually any device networking implementation. This is accomplished because the gateway system enables the capabilities of an embedded device to be exported in the form of a network object. This object can then be integrated into network-based applications. Finally, the future of networked embedded devices lies in a collaborative effort to enable any device, in any industry to communicate over any network.

# # #



06:15:13 PM 01/26

```
  January 26,   6:13 PM

Button Lock
  January 26,   6:13 PM

Mike Howard
  January 26,   6:14 PM
```

4.98v

Figure 1 – Weiser Lock Interface

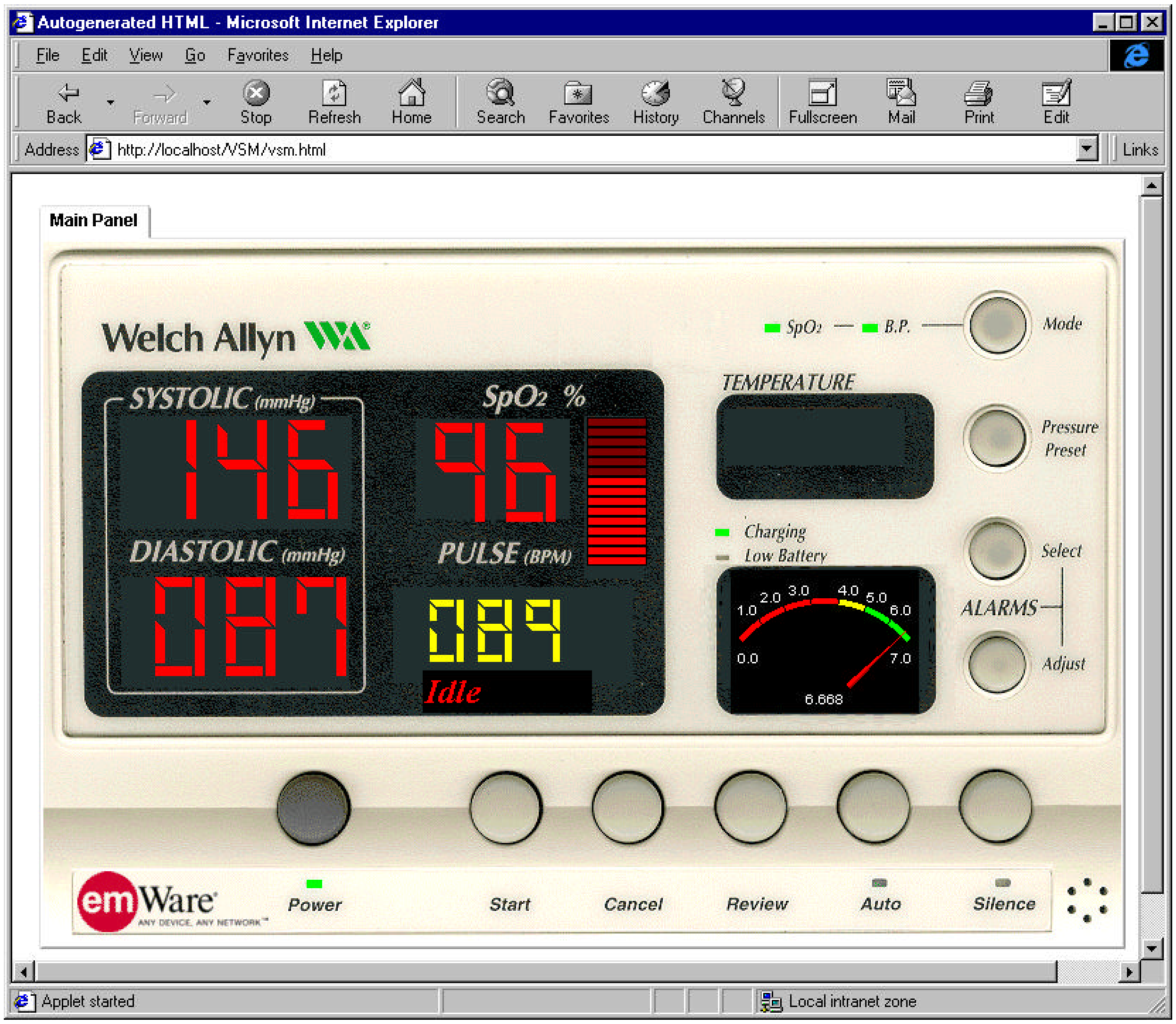


Figure 2 – Welch Allyn Medical Device