 By Edward F Steinfeld, Automata International Marketing

UPnP DEFINES COMMON PROTOCOLS AND PROCEDURES TO GUARANTEE INTEROPERABILITY AMONG NETWORK-ENABLED PCs, APPLIANCES, AND WIRELESS DEVICES.

Devices that play together, work together

WHAT IS THE STATUS of devices on the Internet? And how do you start operations on them? Will the devices notify you when they have a significant event? And is there a simple human interface for the networked devices? The network UPnP (Universal Plug and Play) technology, which Microsoft and the UPnP Forum are promoting, might help you find the answers to these questions.

UPnP technology applies to proximity networks in homes, small businesses, or commercial buildings. For example, the network in **Figure 1** could be in a hospital lab, connecting a cell counter, chromatograph, and blood analyzer; in an office, connecting a print server, fax machine, DSL router, and HVAC controller; in a factory, connecting a PLC, machine-tool controller, and conveyor-belt controller; or in a process plant, connecting setpoint controllers, smart sensors, and a PLC.

The UPnP smart devices in a local network connect using TCP/IP and communicate with a UPnP control point. The control point could be an operator's station or just another device on the network.

In a typical scenario, a network supervisor connects a laptop either directly to the hub, via 802.11b, or remotely across the network to determine what devices are running and the status of each. The laptop is a UPnP control point; it can query the network devices for status and receive transmissions from each device that include a description of the device and a pointer to the device's URL. Better yet, when an event occurs on an UPnP-enabled device, the device notifies subscribing control points of the event.

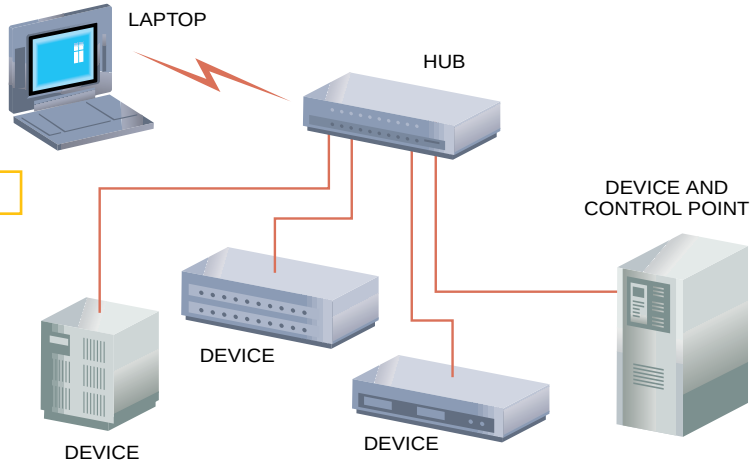
UPnP architecture targets peer-to-peer networking of intelligent appliances, wireless devices, and computers of varying form factors. UPnP defines a set of common services (protocols) that devices can use to join a network and describe themselves and their capabilities, enabling other devices and people to use them without a complicated setup or configuration.

UPnP has six basic layers or functions: device addressing; device discovery; device description; action invocation; event messaging; and presentation, or human interface. UPnP is a protocol for data transmission; it does not move byte codes or use ActiveX controls. It is OS-independent but is built on various network standards. UPnP is designed to work in a peer-to-peer or ad-hoc network. Devices can use a DHCP (Dynamic Host Configuration Protocol) server or Auto IP (Internet Protocol) to automatically choose an IP address from a range of addresses.

DEVICE DISCOVERY

When you add a device to a network, the device advertises itself, letting all control points know that the device is online. A control point can search the network for devices (**Figure 2**). Once a control point

Figure 1



When you add a new control point (in this case, a laptop) to a network, the device may ask the network to find UPnP-enabled devices. These devices respond with their URLs and device descriptions.

The next UPnP Forum meeting will be Oct 18 to 19, 2001 at the Microsoft Corporate Conference Center, Redmond, WA. For information e-mail: upnpevnt@microsoft.com.

discovers a device, the device provides a description of itself and the services it provides. It presents this information in an XML document. Control points can initiate an action on the device by sending a message to the device. For example, the control point may instruct the device to send a message directly to an Internet URL. Once the action completes, the device notifies the control point.

A device does not often need to be told to perform its function, but it does need to report its status to one or more control points. In UPnP, this reporting is called "event messaging," and it is based on a basic push model. Devices send events only to control points that subscribe to a device's events.

Each device can have a unique presentation, or Web home page. It sends the home-page URL to control points as part of its description.

Today, UPnP control points are available only in Windows Me systems. All standard Windows Me systems come with UPnP software, but users must install it. Intel provides a free software-development kit for creating a Linux-based control point. A network can have any number of vendor-enabled or Windows Me control points.

UPnP devices have embedded functions called "services." Services can include turning off the device, scanning inputs for data, or similar functions. Either the device itself or a control point initiates a service. Some objects on a network can be both a control point and a device. This situation is most common in factory systems.

UPnP-enabled devices or control points have six layers of functions (**Figure 3**). Layers 0, 1, and 2 are fundamental; they exist in all devices and control points. Layers 3, 4, and 5 are optional. Control points can initiate an action on a device (layer 3). Many devices offer only event messaging (layer 4); the device creates an event, and a control point listens for the event. Devices can send data or the results of an action they have taken without initiating a control point. Some devices might provide a control point with only a presentation user interface (layer 5). The control point's browser displays the device's Web user interface. This user interface may either display events or status or control the device.

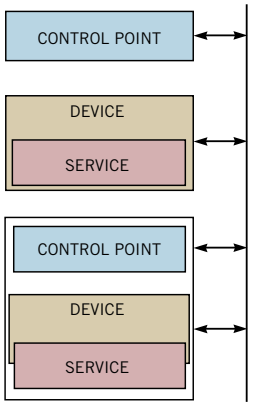


Figure 2 UPnP-network nodes can be devices, control points, or both. Devices have services that a control point can monitor or control.

The presentation layer appears to be required, because a pointer to the presentation URL is part of the device description. However, if the control point is handling the device programmatically and not through a Web browser, the presentation layer is not required. Because every UPnP device uses a Web server, it is simple to use the control point's browser as the front panel of the device.

The addressing layer is where control points and devices obtain their IP addresses. The addresses can be hardwired or come from a DHCP server, or devices can use Auto IP to assign an IP address. Auto IP is a new draft of the IETF (Internet Engineering Task Force) standard, Dynamic Configuration of Ipv4 Link-Local Addresses. Auto IP permits a UPnP device to select an IP address from a range of nonroutable addresses. Once selected, the device tests the address to determine whether another device is using it.

YOU SAY HTTPU, I SAY HTTPMU

Searching for devices on a network or advertising that a device is on a network is the function of the discovery layer. When you add a device to a network, the discovery layer advertises the device's presence by sending a multicast variant of HTTP (HTTPMU, or HTTP Multicast UDP). A control point answers using a unicast variant of HTTP (HTTPU, or HTTP Unicast UDP). HTTPMU is a new addition to the HTTP standard. In **Figure 1**, a control point (the laptop) is added to the network. To find out what UPnP nodes exist, the control point sends out a HTTPMU, and each UPnP-enabled device responds with an HTTPU reply.

Once a control point discovers a device, it can obtain a description from the device. The device expresses the information in XML (Extensible Markup Language). XML is used throughout the UPnP implementation. A description includes a device type, URLs for control and events, icons, and a URL for presentation, as well as the manufacturer's name, serial number, product code, and other similar information. The UPnP Forum defines device types. Each device type may have one or more UPnP templates to define the content and presentation of data.

To initiate device action, a control point sends a control message using the definitions from the device-description document. To exchange information, the control layer uses a W3C (World Wide Web Consortium) draft standard, SOAP (Simple Object Access Protocol). The consortium defines SOAP as a "lightweight, XML-based protocol for exchange of information in a decentralized,

UPnP STILL NEEDS WORK

Today, no templates or standards dictate how industrial machine tools, automatic test equipment, or medical devices will communicate data using UPnP (Universal Plug and Play). UPnP working committees create device-specific templates. A number of templates exist, but they are mostly for consumer devices.

Invensys, an automation-and-controls-industry vendor, has asked the UPnP Forum to create an industrial working committee to define templates for factory and process-plant use. Templates provide interoperability rules for manufacturers of similar devices with control points. You can still implement UPnP on the factory floor without the comfort of a common standard of data identification. In fact, many devices support discovery and presentation without the use of standard templates.

There has been no testing to determine the number of UPnP devices a network can support. You can imagine that after a power failure, every active UPnP device will send multicast event notifications, and every control point will send inquiries of discovery. For a network with many UPnP-enabled devices, such traffic could bring the network to its knees.

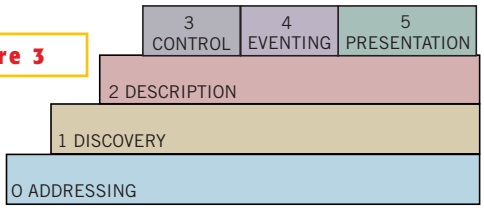
distributed environment." The device completes the action and responds using SOAP.

The event-messaging layer appears to be a basic push model in which control points listen for notifications of UPnP-device state changes. But actually, the process is a little more complex. To obtain event messages, control points subscribe to event messages for a specific service within a device. A network can contain multiple control points and multiple UPnP-enabled devices. A control point might want to listen to more than one network service but not all of them. By subscribing and unsubscribing to events, control points can be selective. When a service within a device has an event, it sends that event (Notify) to all current subscribers. Consequently, all subscribers have current knowledge of the state of the device.

Event messages use GENA (General Event Notification Architecture), an extension to HTTP defined by the IETF draft standard. GENA is designed to send and receive notifications using HTTP over TCP and UDP.

The presentation function presents information and control to the user. Presentation requires the completion of layers 0, 1, and 2: getting an address, discovering the device, and obtaining the device description. The description document provides the device URL for the presentation page. There are no constraints on the use of the Web page you obtain from the device. The device-description document provides a URL for the initial presentation page for the device. An unlimited number of linked presentation pages is permitted.

Figure 3



UPnP's six layers consist of IP addressing; discovery; description of URLs and services; optional control of other UPnP devices; event messaging; and presentation, or the Web page for the device. Layers 0 to 2 exist in all UPnP-enabled devices and control points.

PLUG-AND-PLAY STACKS

So far, UPnP looks straightforward. However, some pieces of the technology that UPnP uses are absent from typical network-enabled devices. Many implementers have no knowledge of some of the newer protocols.

Most implementers will purchase their UPnP protocol stacks as they now purchase TCP/IP stacks. The descriptions of the functions mention some of the protocols that make up UPnP. You can find complete information at the UPnP Forum, W3C, and IETF Web sites.

UPnP starts with IP (**Figure 4**). UDP is used for discovery and events because it is multicast. TCP handles description, control, and presentation. A device uses HTTPMU to broadcast its presence to the network, and a control point uses HTTPMU to ask what devices are present. Both use GENA and SSDP (Simple Device Discovery Protocol). The control function uses SOAP. The IETF define GENA and SSDP, and the W3C defines SOAP. HTTPMU is not part of the HTTP standard. HTTPMU was created for use in UPnP, and HTTPU is used in the response to a search as well as in description, control, and events.

The UPnP Device Architecture layer, the focus of this article, defines the UPnP structure. Microsoft designed the architecture and contributed it to the UPnP Forum to be used royalty-free. Working committees of the UPnP forum define UPnP Forum Templates. They also define domain- and device-specific meaning and data format on top of UPnP Device Architecture. UPnP vendors may also specify their own extensions to working-committee definitions. Vendors can use any language to call UPnP functions. UPnP is designed to be both language and operating-system independent.

The UPnP vendor-specific layer comprises the application, the user interface, and vendor-specific hardware. Vendors can deliver UPnP on a variety of physical networks; the only requirement is that the network must support IP.

You can obtain more information at the UPnP Forum Web site (www.upnp.org). To obtain complete information, you must first join the UPnP Forum, which charges no membership fee. The UPnP Forum, a nonprofit organization, defines UPnP protocols and UPnP device templates and is the source of device certification.

The UPnP Forum is an industry initiative designed to enable easy and robust connectivity among stand-alone devices and computers from many vendors. The UPnP Forum is open to any company wanting to participate in driving the adoption of UPnP. Companies with interests in particular device classes should

FOR MORE INFORMATION...

For more information on products such as those discussed in this article, go to www.ednmag.com. When you contact any of the following manufacturers directly, please let them know you read about their products in *EDN*.

Allegro Software Development Corp
1-978-264-6600
www.allegrosoft.com
Enter No. 301

IETF (Internet Engineering Task Force)
www.ietf.org
Enter No. 302

Intel Corp
1-408-765-8080
http://intel.com/ial/upnp
Enter No. 303

Invensys Software Systems
1-703-234-6000
www.invensys.com
Enter No. 304

Metro Link Inc
1-954-660-2500
www.metrolink.com
Enter No. 305

Microsoft
www.microsoft.com
Enter No. 306

Mitsubishi Electric and Electronics USA Inc
1-408-730-5900
www.mitsubishichips.com/products/mcu/index.html
Enter No. 307

NetSilicon Inc
1-781-647-1234
www.netsilicon.com
Enter No. 308

UPnP (Universal Plug and Play) Forum
www.upnp.org
Enter No. 309

W3C (World Wide Web Consortium)
www.w3.org
Enter No. 310

SUPER INFO NUMBER
For more information on the products available from all of the vendors listed in this box, go to www.ednmag.com, click on the Reader Service link, and enter no. 311

become UPnP Forum members and participate in the process to design schema templates for their device classes. More than 300 companies are UPnP Forum members.

SOFTWARE-DEVELOPMENT AND TOOL KITS

Both Microsoft and Intel offer UPnP software-development kits at no cost to developers. The Intel Linux SDK is available online at http://intel.com/ial/upnp or http://upnp.sourceforge.net, and the Microsoft SDK for Windows development is available at http://www.microsoft.com/hwdev/upnp.

The Intel Linux SDK includes API and Linux source code to implement UPnP-compliant control points and devices, sample source code with comments, header files, an integrated Web server, an optimized XML parser, documentation, and a Berkeley-style open-source license. The Microsoft SDK includes code for discovery, control, and events; Windows 2000 and Windows CE support; ISAPI (Internet Server API) control, which works with an IIS (Internet Information Server) or Windows CE Web servers; a mini-XML parser; documentation; tools, such as generic UCP and network-monitor parsers, and sample code that implements the X10 light bridge.

If you are unfamiliar with GENA, SSDP, SOAP, and multicast HTTP or if you would rather have someone else do all the basics, a small number of vendors are shipping UPnP tool sets. The tool-set vendors are members of the UPnP Forum. They have tested their products at UPnP Plug Fests for adherence to the specification and compatibility with other vendors' products.

Products shipping today from Allegro Software Development Corp include RomUPNP Basic and RomUPNP Advanced tool kits. RomUPNP Basic includes discovery, description, and presentation, and RomUPNP Advanced adds events and control. Control turns a device into an UPnP control point. These products, along with all other Allegro Software products, such as its micro-XML parser and Web client, follow the UPnP principal of being hardware, operating-system, and language independent. Allegro provides a UPnP Device Viewer program for all Windows systems that looks for UPnP devices on the network and provides a list in a window. You can click on the device name to see a complete device description or double click to launch a browser and display the device's HTML page. When a UPnP device is shut down (not aborted) or a new UPnP device comes online, it will be listed in the control-point window.

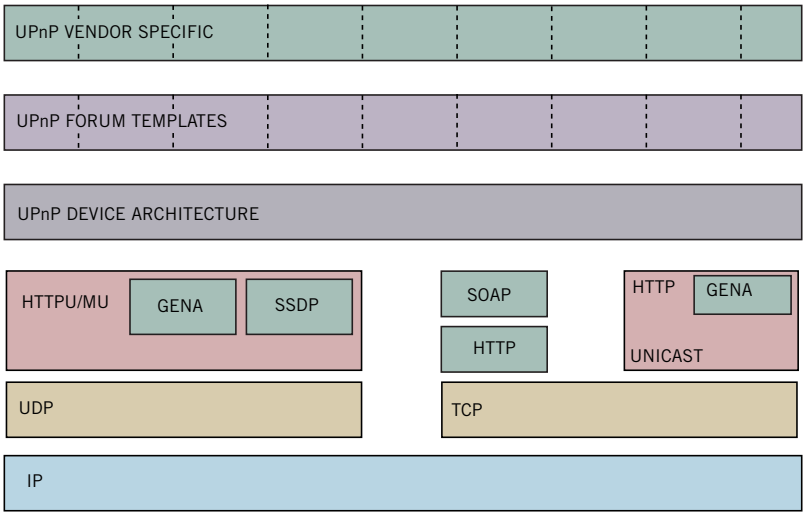


Figure 4 UPnP-network-device implementers use protocol standards, such as GENA (General Event Notification Architecture), SSDP (Simple Device Discovery Protocol), and SOAP (Simple Object Access Protocol), to enable automatic discovery and description.

WHAT ABOUT JAVA?

For companies using a total Java option, Metro Link Inc has a complete set of Java-based UPnP software stacks. The Metro EnableWorks Device software-development kit for Java was built for both Java 1.1 and 1.2. It works with any OS that is fully compliant with Java 1.1 or higher. The Metro EnableWorks UPnP control-point software-development kit for Java gives developers all the tools they need to develop UPnP control points for Java-1.2-compliant platforms.

Silicon is coming to UPnP in 2001. Mitsubishi Electric and Electronics has announced a set of microprocessors for the implementation of Internet-enabled devices. Mitsubishi will include a Control Point reference design with these offerings.

NetSilicon will be delivering UPnP software along with its NET+ARM system on a chip, a high-performance, highly integrated embedded Ethernet processor targeting today's demanding network-appliance applications. NetSilicon believes that UPnP will help untrained workers to use network-enabled devices.

As more devices become UPnP-enabled and more workstation OSs become UPnP control points, it will be easier to deploy smart devices on the network. A workstation's browser will become the operator interface for all the smart networked devices. Operators can start actions in the devices, or the actions can begin programmatically, and operators will be able to obtain the results the same way. The best part of UPnP is that it lets you find out what devices are on the network and how to communicate with them.

UPnP implementation is not rocket science, and excellent tools are making it even easier to design UPnP-enabled devices. User-interface designs can be as simple as HTML pages, with no additional hardware required. Once vendors and users see the power of UPnP, every network-enabled device will be UPnP-enabled.□

AUTHOR'S BIOGRAPHY

Edward Steinfeld has more than 25 years experience in real-time and embedded computing. He has marketed embedded and real-time products to OEMs and resellers for Digital Equipment Corp, VenturCom Inc, and Phar Lap Software. His international experience includes a stint in Hong Kong as a Far East channels manager and responsibility for international OEM marketing in Europe and the Pacific Rim. Steinfeld is now providing market research, planning, and services to the embedded-computing industry. He has been an evangelist for embedded Web products since 1995, when he announced the "World's Smallest Web Server" for Phar Lap Software. At the time, it may have been the first diskless Web server.