Person to Equipment Communication

Tools and Technologies

Konrad Froitzheim
Technical University of Freiberg
Germany
mailto:frz@tu-freiberg.de

Preface

An important part of virtual laboratories are experiments jointly and remotely used by the researchers. We will give a short taxonomy of Person to Equipment communication (P2E) before we discuss the components of such a remote control experiment: network-enabled transducers, communication technologies, Webserver extensions, abstract instrument description technologies, and result transport mechanisms. At the end we will give a few examples. The engineer about to build a remote experiment should understand this document as an overview on the architecture, the problems and the technologies of person to equipment communication.

1 Virtual laboratories

It is not the intention of this paper to develop a taxonomy of virtual laboratories. We should however give an idea for such a taxonomy:

- *a large scale facility with remote access;*
- *a network of tools or of laboratories;*
- *a network of scientists, characterized by a clear membership to a given community (such as expatriated national scientists).*

Based on such an understanding of a virtual laboratory, it is clear that communication tools are at the heart of such an undertaking, virtual laboratories spanning multiple institutions are usually geographically distributed. They are also heterogeneous in terms of computing and communication equipment.

Considering this short VL-taxonomy, one arrives at two major communication tool classes (compare [Fluckiger]):

- person to person communication in a network of scientists
- person to equipment communication to control a network of tools

2 P2E scenarios

An important part of many virtual laboratories are experiments. They can be operated by certain manipulators and the results are collected by measuring equipment, which can again be controlled. In virtual laboratories this operation and control can be performed remotely.

The control of the equipment can either be performed interactively (typically called teleoperation) or asynchronously with a predefined procedure, script or program (teleprogramming). Depending on the method chosen, synchronous feedback to the scientist may be necessary.

2.1 Teleoperation

In the teleoperation scenario a scientist gives commands to remote equipment. The equipment is typically a measuring device (telescope, microscope, camera), a manipulator, or a probe. These commands are typically of a 'strategic' nature (move to position x, fill tank, explode, etc., see figure 1). Fine control will be performed by the equipment itself, which also prevents catastrophic behavior. In this case the feedback channel (mostly video or sample streams) is used to inform the remote scientist of the status of the system and whether a strategic goal has been achieved.

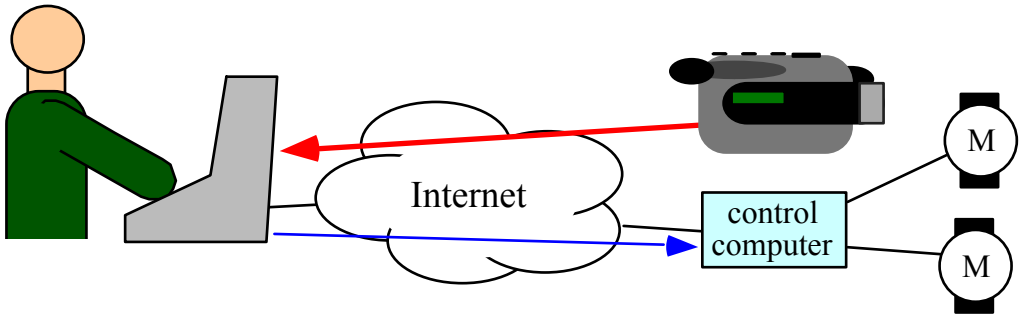

Figure 1: Operator-controller-equipment. The motors on the remote site are controlled by the operator based on camera feedback. The blue line represents the control stream, the thick red line symbolizes the high volume video stream. Physical connections are drawn as a fine black lines.

In the second mode the commands are on a lower level (move right, pour fluid, stop, see figure 2). In this mode the feedback channel is of utmost importance, since very high interactivity is required. The critical nature of the feedback imposes high quality of service requirements on the communication channel with respect to delay and throughput.

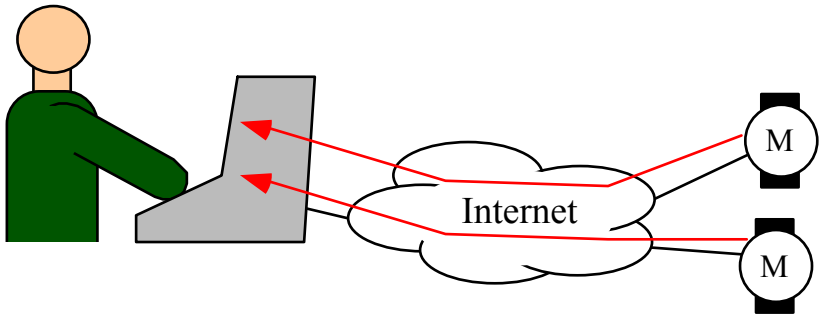

Figure 2: Operator-motor-equipment. Feedback is generated by sensors in the motor and delivered in real-time to the operator. The red lines represent time-critical, low data rate feedback streams. Physical connections are drawn as a fine black lines.

The result of the experiment, i.e. the data (media) stream obtained can either be collected and stored at the equipment site for later transfer, or it can be transmitted live to the user site. The second is mostly the case when the feedback channel also contains results.

Application sharing [Timbuktu] can be used as a tool for teleoperation. The local computer based equipment control program to operate the experiment will in this case be used remotely. The application sharing service transmits the user interface of this program to the remote site and the remote input is fed back into the program. Application sharing systems include WTS (Windows Terminal Server (Microsoft), Citrix), Proshare, and Timbuktu from Netopia.

WWW-based control interfaces are implemented with a standalone Webserver augmented by plugins, extension modules, and/or CGI-programs to communicate with the instrument or by integrating a small Webserver into the instrument. The user-interface is then based on simple or advanced html-pages with certain embedded links pointing to the equipment control software complete with parameters. The Internet Model Railroad, WebIR - a WWW-based infrared remote control for VCRs, or various WebCams are examples for this technology.

In his excellent book on remote control robotics Craig Sayer writes [Sayer]: "The difficult part of remote operation is not the transferal of commands to the remote site, nor is it

the interpretation of those commands as physical actions. Instead it is the development of mechanisms to detect and overcome those problems that will inevitably occur." As in software engineering the problem does not have a prefabricated solution, not even a theoretical solution. The only help is to design and implement many safeguards and recovery mechanisms. The need to work despite mechanical failures has been a major part of the control software for the Internet Model Railroad (see below).

2.2 Teleprogramming

Teleprogramming is an asynchronous approach to the operation of equipment in a virtual laboratory. The scientist creates a series of commands for the device which is then downloaded into the device and executed. The result stream is recorded and later sent back for evaluation by the scientist.

This command series is in many cases either a script or a program. Most programming languages can be used as long as the equipment manufacturer supports them. Today Java is certainly a good choice, as long as the programs created are not time-critical. Two problem should be mentioned in this context:

- Programming is a task intimately associated with errors (bugs). Since the remote programming task is especially tedious with respect to turnaround times, local simulation of the programs is very important. Such a simulation environment helps reduce the time needed to parameterize and launch the program and can avoid catastrophic errors. Simulation environments for experiments and equipment are, however, not very common.
- In order to write a program for a given experiment, a formal description of the functionality of the equipment and of the controllable parameters is a prerequisite. This can be in the form of interface files or distributed programming interfaces (CORBA, RMI-objects, etc). Again not every equipment manufacturer supports this.

Significant research on teleprogramming for robots bas been performed 1990-1992 by Paul, Sayer, Funda, and Lindsay at the University of Pennsylvania [Funda, Sayer].

3 P2E Components

3.1 Transducers

Every experiment needs transducers, i.e. actuators and sensors. For the purpose of virtual laboratories, they have to be connected to a long distance network, usually the Internet. Options to embed the control software, commands, and feedback streams of the transducers into Web-based mechanisms are discussed in chapter 3.2.

3.1.1 Structure of a networked Transducer

Although sensors and actuators are conceptually separate entities, most practical actuators have built-in sensors (e.g. an RPM-regulated motor) and sensors have integrated actuators to control their operation (e.g. mode selectors). The task to design and build cost-effective, networked transducers for highly diverse tasks has been taken on within a

standardization effort, IEEE P1451, by the National Institute for Standards (NIST, USA) and the IEEE [Lee]. The efforts are currently (summer 2002) in various states of the standardization process:

- IEEE P1451.1, the *Network Capable Application Processor (NCAP) Information Model* will define an object model for the components of a smart transducer (right pink box in figure 3) and the software interface specifications of these components. Encapsulation of the hardware implementation and the protocols used to access the NCAP are the high level solutionconcepts. C and C++ implementations have validated the concepts, so that the standard is close to completion.
- IEEE P1451.3 (*Digital Communication and Transducer Electronic Data Sheet (TEDS) Formats for Distributed Multidrop Systems*) introduces a bus system, so that several transducers can be served by one NCAP.
- IEEE P1451.4 *(Mixed-mode Communication Protocols and Transducer Electronic Data Sheet (TEDS) Formats)* defines an operation mode in which digital communication data (TEDS) is multiplexed with analog sensor data on a single wire pair.
- IEEE P1451.2 (*Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet (TEDS) Format)* is as of summer 2002 the only full standard. It specifies a transducer-to-network processor hardware and software interface (the red line in figure 3) and a Transducer Electronic Data Sheet (TEDS, the pink box on the left of figure 3).

  The interface part of P1451.2 specifies elements such as register access and addressing. The TEDS is the part of IEEE P1451 which is most relevant to the developer of P2E communication tools. Significant work has been done to specify equipment properties and especially on the coding of physical units (see below in the chapter on abstract equipment description).

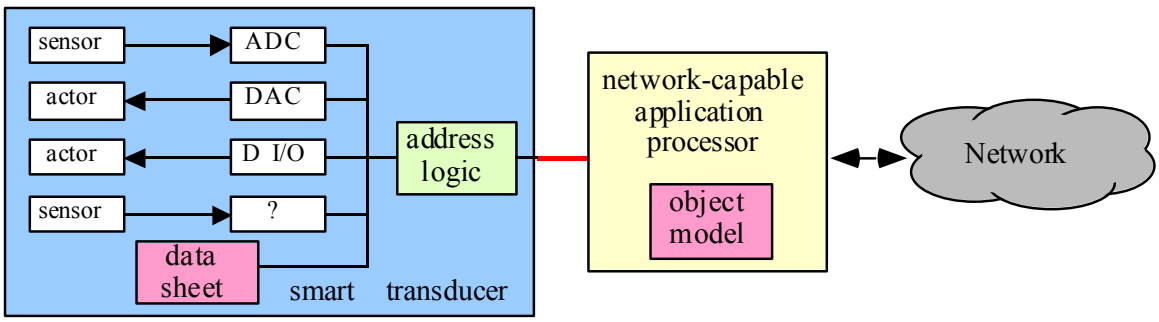

Figure 3: Smart Transducer as introduced in the IEEE P1451 standard.

3.1.2 Actuators

In the context of this document actuators are the parts of an experiment which purposefully manipulate either the object to be experimented with or the measurement instrument(s). Examples of actuators with different levels of complexity are:

- Switches are actuators with discrete values. Value examples are 'on', 'off', and the parameters of program selectors, mode switches, gears, menu selections, etc. Switches are set to a value and the switch actuator holds that position until the next setting is sent to the switch.
- Dials can assume conceptually continuous values, i.e. they are actuators that set parameters on a non-discrete scale. Examples are the tuning of a radio, setting the voltage of a lab power supply etc.
- Motors fall in both classes: they can be either of the pseudo-continuous kind (select the rpm) or discrete, such as stepper motors.
- The highest level actuator is a robot. Robots can execute tasks such as 'move to position', 'transport item', 'insert probe'. These tasks are a sequence of simple actions. The execution of such a sequence is typically under the control of feedback clauses.

A surprising observation is that these elements are strikingly similar to the basic elements of Human Computer Interfaces (HCI). HCI research has long dealt with these elements. In computer interfaces the actuators are called switches, radio buttons, menus, dials, sliders, or scrollbars. After some thought this fact seems less surprising, an experiment is a complex machine like the computer.

This observation opens the opportunity to apply the well explored concepts of HCI and programming APIs to the field of remote control actuators.

3.1.3 Sensors

Scientific sensors produce two categories of data streams:

- Discreet numerical output reflects the value at the sensor state at a single point in time. It is typically easy to transmit since it is not time critical. Date, time and calibration information are often required components of the data set.
- Continuous numerical data is actually sampled, resulting in a stream of discrete numerical output. The sampling rate can range from one per day to microsecond intervals. Streams with higher sampling rates can be hard to transmit over the Internet due to its unpredictable QoS (see below). Another problem with this kind of sensor output is the sometimes enormous amount of data generated.
- Multimedia output includes still pictures, sometimes audio, and often video. [Halsall] gives an excellent overview of media properties, formats, and communications protocols for this class of sensor output. We will look into still pictures and video sensors and distribution software in a later chapter.

3.2 Internet-Technology Interface

Internet Technologies in the sense of this chapter are hardware such as networking components (Ethernet, Modems, Routing, etc.), protocol implementations (TCP, IP, PPP, etc), standard software systems (webserver, database, etc.) and data formats (HTML, XML, electronic datasheets, etc.. During the discussion we will of course try to avoid repetition of common knowledge, but some basic facts need to be recalled:

3.2.1 Internet and Quality of Service

Quality of service is a set of performance parameters associated with a certain service, especially with data transmission services. Typical parameters are:

- throughput, the amount of data (packets, bits) transmitted time unit [packets/sec]
- delay, the time a packet needs from entering the network to delivery [seconds]
- delay jitter (variations of the delay)
- error probability (lost packets, bit errors)
- error detection probability (effectiveness of checksums)
- error correction measures (retransmission, forward error correction)
- topology (unicast, anycast, multicast, broadcast)
- availability (probability of a successful connection set-up)

A given network may always provide certain values of these parameters (guaranteed QoS), achieve them on average over a given time (statistical QoS), or just strive to maintain the values (best-effort) QoS. Guaranteed QoS is for example provided by the ISDN telephone system regarding data rate, but not in all the other areas.

The publicly available Internet is based on the best-effort QoS paradigm. In reality Internet-QoS is completely unpredictable. Although research and development is trying to add true QoS to the Internet (IntServ, DiffServ), reliable QoS in today's Internet is based on overdimensioning the system (links and routers).

This creates the necessity to design the remote experiment such that it survives changes in QoS and to use robust media transmission services. We will discuss such services later in this report.

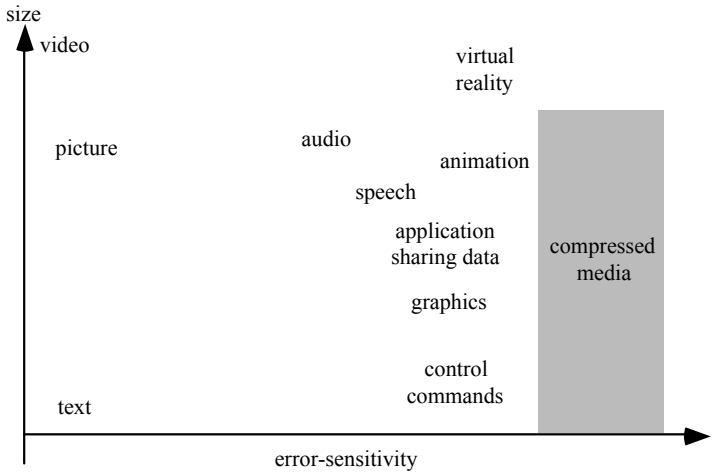

Figure 4: Media stream size versus error-tolerance

3.2.2 Embedding a Webserver into the device control software

If an instrument is controlled by software through a computer and that software is available as source code, it is often a good option to embed a very simple Webserver into the software. The fact that modern Webservers (e.g. Apache, IIS) are complicated and big is

due to their extensibility, elaborate load balancing options, optimizations, logging, and access rights management. An instrument however does not need the mechanisms that make full size Webservers bulky.

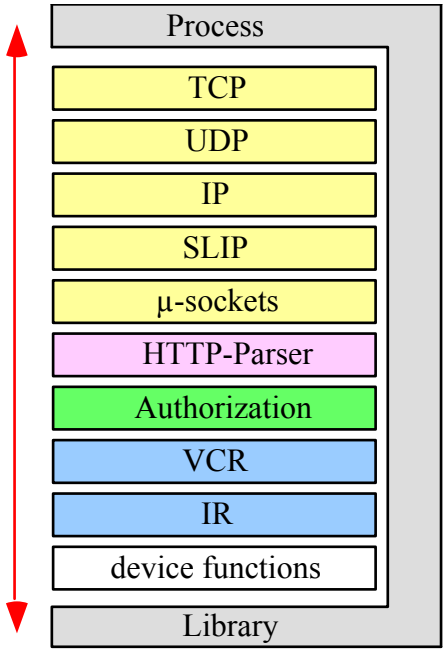

Figure 5: Structure of the Internet enabled Infrared Remote Control

An instrument Webserver can focus on embedding and evaluating interaction elements into a few html pages. The other important element of this server is access control. This leads to a compact software module that typically consumes only a few hundred lines of code. As an example the http modules (i.e. Webserver) for a Web-enabled infrared-remote-controller [Birkenmaier] was only 780 lines of C that compiled into 10.8 Kbytes of code (see figure 5).

3.2.3 Extending a Webserver - PlugIns, Modules, Filters

Webservers are readily available for most operating systems. Microsofts Internet Information Server IIS and Apache are widely distributed servers with an extensible architecture. Apache islicensed as open source software (apache.org), IIS comes with every Windows server license. A computer capable of running these Webservers can be used to interface to the virtual laboratory instrument. Depending on the instrument capabilities and the complexity of the control task, several options exist to extend the standard Webserver to control the instrument:

- Scripts embedded in html documents (php, asp)
- Custom modules (ISAPI .dll or an Apache module)

3.2.3.1 Apache Modules and php

As of version 2 'external' modules can be executed by the Apache Webserver (see figure 6). Such a module can be implemented to control an instrument . In this case

developers have basically all the operating system functions in the computer at their disposal to write the software.

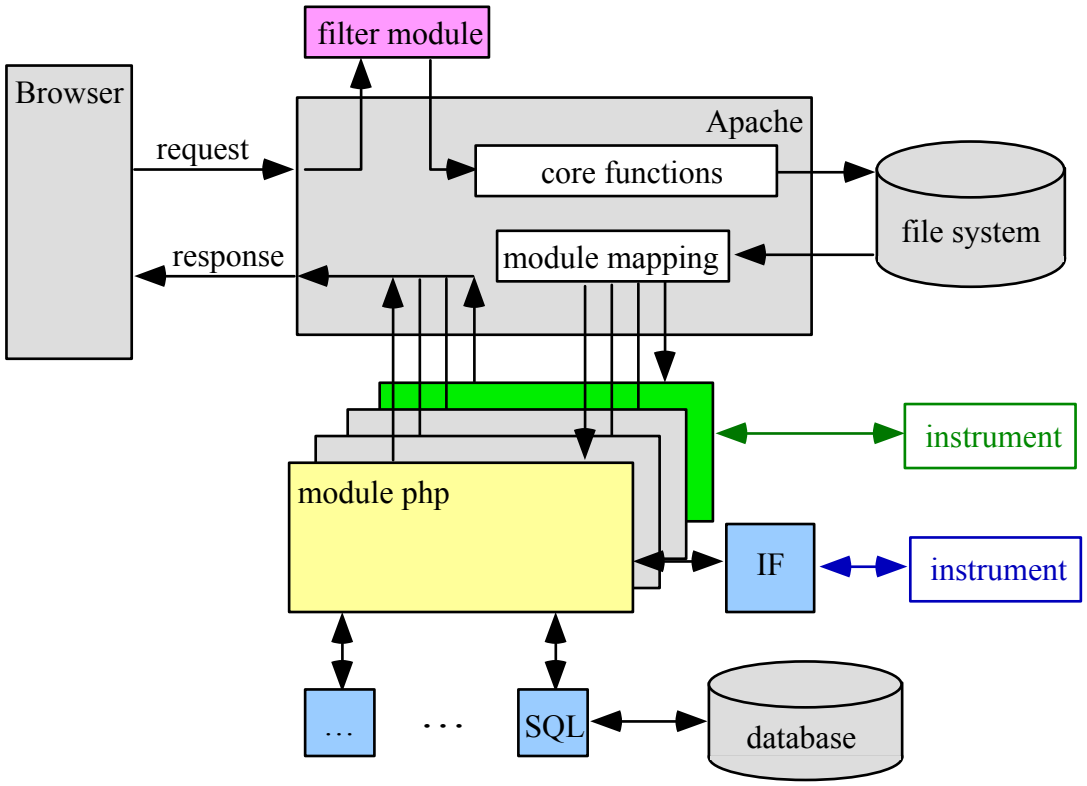

Figure 6: Apache extension architecture: a filter module transforms the incoming request form the client. After the Apache core functions process the request and retrieve the requested object from the file system, a mapping mechanism sends the object to an Apache module. A custom module (green) could be implemented to control the instrument. Another option is to use mod_php (yellow) to execute a php script interfacing to the instrument.

An existing module is module_php. It processes php scripts embedded in an html document. Many books exist on the subject, a tutorial can be found on the php.net Website [phpman]. The fundamental idea is that mod_php takes the html document, looks for php scripts, executes them, and embeds the result into the html document (compare table 1).

Before the execution of the script	After the execution of the script
<html> <head> <title>PHP Test</title> </head> <body> <?php echo " Hello World "; ?> </body> </html>	<html> <head> <title>PHP Test</title> </head> <body> Hello World </body> </html>

Table 1: A trivial php example. The page stored on disc is shown on the left, the page returned to the client is on the right.

Php scripts typically make intensive use of functions built into the php environment (see [Phpfunctions]). Examples are functions to access SQL databases (e.g. postgres, Oracle,

MySQL), to parse XML or XSLT, to access sockets directly, interface to Windows COM modules, and many more. Using php for intense calculations such as signal processing on result data is not a good idea, however: php is interpreted at runtime, so e.g. numerical performance is insufficient.

3.2.3.2 IIS, ASP, and ISAPI

Microsofts Internet Information Server IIS offers similar extensibility, although the architecture of the extension mechanism is different. IIS can be extended with Active Server Page (asp) scripts. These scripts typically invoke COM objects and the results are then placed into the outgoing html document. The mechanism resembles the php information flow, although the functionality seems more powerful because COM objects basically support the whole Windows functionality.

An ISAPI filter can pre-process the incoming request and, after the document is retrieved by the regular IIS mechanisms, IIS can invoke Internet Services API (ISAPI) extensions (Windows dynamic link libraries, dll) for the document. Similar to Apache such an ISAPI dll can be written in many different programming languages (typically C or C++) to access operating system functions (typically through the MFC). Microsoft Developer Studio even includes a wizard to generate an ISAPI project with all the interface code.

An ISAPI-dll for php can be downloaded from php.net, so that php can even be used in the framework of IIS.

3.2.4 Abstract equipment description

Common to both P2E implementation scenarios (teleoperation and teleprogramming) is the fundamental task of a equipment description, so that applications can be designed and software can be written for general equipment control rather than only for a specific instrument in a particular experiment.

Another good reason for abstraction is given by Sayer [Sayer]: "… overcoming bandwidth constraints by communicating at a more abstract level - is fundamental to remote control via constrained communications".

In order to simplify remote experiment creation and usage, standards and descriptions have to be used and/or created to:

- find actuators and sensors
- control the actuators
- control sensors
- receive data streams from the sensors

3.2.4.1 Instrument properties

As we motivated above, properties, interaction parameters (values, modes, etc), and measurement data of the instruments (sensors and actuators) should be described in what some authors call an 'Electronic Datasheet' (see IEEE P1451 TEDS). Such a datasheet should contain this information in a common structured format, which can be interpreted

by standard software (such as a Web browser) or by a standard application for remote control (TEDS falls in the latter area).

One major area where standardization is badly needed and surprisingly not too difficult is the representation of physical values,i.e. the flexible coding of numbers and the specification of the physical unit associated with each value. For physical units, the SI system (Système International d'Unités [bipm]) is of course the first choice. IEEE P1451.2 specifies a solution. The high-level structure of an IEEE P4151.2 TEDS is given in figure 7. [Woods] gives an enlightening example for TEDS in the appendices of his paper.

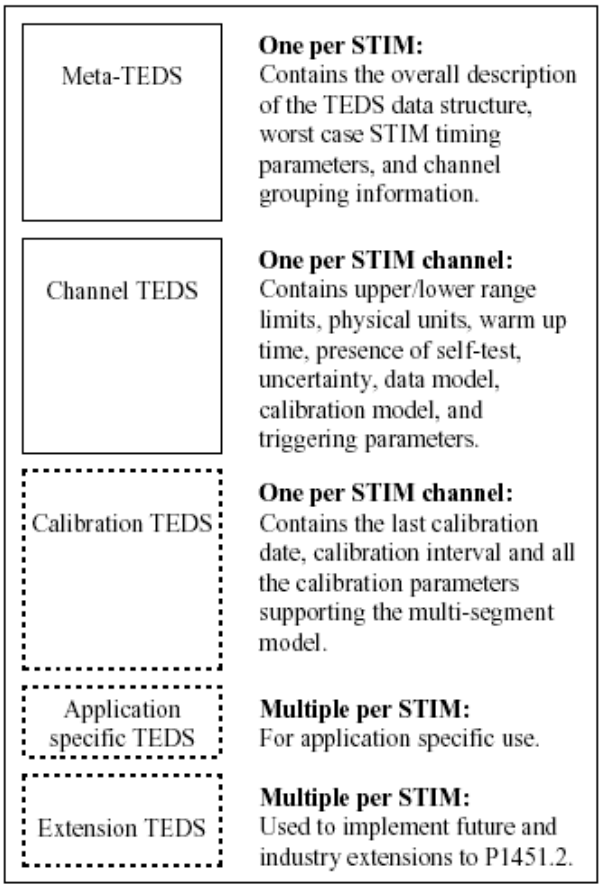

Figure 7: Structure of the IEEE P1451.2 electronic data sheet. STIM means Smart Transducer Interface Module. From: [Woods]

An issue not sufficiently addressed in IEEE P1451 are more complex data structures produced by many sophisticated research instruments.

The eXtensible Markup Language **XML** [xml.org] is a recent W3C standard based on the well established Standard Generalized Markup Language (SGML). Because the ubiquitous HTML is also closely related to SGML, those who can read HTML find XML files very familiar. The true difference is in the semantic of the language. Whereas HTML was developed to represent *text* in such a way that it can be formatted by a browser (mark up), and allows for a reasonable hypertext user interface, XML was designed to represent *data*

in an extensible way. And that is exactly what is needed to describe properties and results of equipment in virtual laboratories.

XML itself does not have the built-in text-rendering semantics of html. XML is used to describe data for transport and processing. The user (programmer) is responsible for the description of the datatypes. This description, a namespace, is specified in XML and either embedded in the document or accessible through a URL given in the document [MSXML].

XSIL, the Extensible Scientific Interchange Language, is an XML DTD (document type definition). It connects scientific XML to a Java object model and a visualization system. It includes an XML format for scientific data objects, and a corresponding Java object model, so that XML files can be read, transformed, and visualized [Williams]. XSIL includes base objects for tables and arrays to represent scientific data sets. The extensibility of XML together with the XSIL object model make it easy to add more complex data types to XSIL. These data structures can be filled from the stream object repersent the actual instrument output.

3.2.4.2 Instrument Control

In order to control instruments over the network, commands have to be sent to the instrument. This can be achieved through:

- a Web-based mechanism. Commands and parameters are embedded in the parameter part of the URL (after the '?') or in a form posted to the server. A server extension module (see above) decodes the command, translates it and sends it to the instrument. This option is used in most simple Web remote control systems.
- a classic RPC (Remote Procedure Call) using a system such as SUN RPCs or DCE. These systems require custom software at both ends. A long standing problem of all RPC mechanisms is the encoding (marshalling) of the procedure parameters. XDR, the eXternal Data Representation language is used in the SUN-RPC case, IDL for DCE.
- XML-RPCs are an emerging standard to send RPCs in XML documents. They use HTTP as the transport and XML as the encoding for remote procedure calls. XML-RPC is designed to be as simple as possible, while allowing complex data structures to be transmitted, processed, and returned [xmlrpc.com]. The advantages of using XML lie in the strong data-encoding capabilities of XML. Many implementations are already available, e.g. Perl, Python, Java, C/C++, PHP, Microsoft .NET, and Zope.For transmission, XML-RPC values are encoded as XML [Kidd]:

```
<methodCall>
     <methodName>sample.sumAndDifference</methodName>
           <params>
                 <param><value><int>5</int></value></param>
                 <param><value><int>3</int></value></param>
           </params>
</methodCall>
```

3.2.4.3 Handling Multimedia Results

The output of an instrument in many cases includes one or more multimedia element(s) or stream(s). The most prominent is certainly video, but still pictures and in some cases audio can also be required. Since simple image and video formats are well known, we will limit ourselves here to an overview of the basic ideas of video capture and transmission in the Web.

Still pictures

The capture of pictures from a camera which are then statically included in a Web page has been part of the Web almost as long as it has existed. The early examples have been the 'Trojan Room Coffee Pot' or the FishCam. Tens of thousands of such WebCams are active on the Web. Most of them are inexpensive digital video cameras (from companies such as Creative, Philips, Connectix/LogiTech, etc.) connected to PCs over a standard interface such as the parallel port, USB, or FireWire (IEEE 1394). Simple software in the PC then grabs single frames at predefined time intervals and copies the resulting JPEG pictures into the Webserver subdirectory with the appropriate filename, so that a static Hyperlink in a page points to the most recent picture. This mechanism includes the latest picture on the page that is loaded into the browser: later updates have to be 'forced' by the user with the reload-button or automatically with the client-pull feature of the browser.

Live Video

The video capture mechanism of live video WebCams is similar to the above scenario. What is considerably more complicated is the software in the server, which constantly pushes the latest frames in the picture stream to the client. Depending on the Web browser (Netscape, Internet Explorer, Opera, etc.) different mechanisms have to be used to enable the continuous display of the video in the client:

- animated GIF (Netscape 3, 4 and certain versions of 6)
- JPEG server push (Netscape 2, 3, 4 and certain versions of 6)
- Java applet (Internet Explorer)

An important element of video streaming over the Internet is compression. The stream formats (and still picture formats) mentioned above (GIF, JPEG) already use compression. More efficient video compression can be necessary if the requirements such as frame rate and resolution are higher. Applicable algorithms and stream formats are MPEG-1, MPEG-2, and MPEG-4. They reduce the bit-rate by almost an order of magnitude, but they require a lot of processing power and introduce significant delay (because of the compression algorithm itself or implementation pipelines). Such a delay can be a problem in control scenarios, where the round trip delay (command-action-feedback) is critical.

Advanced Video-over-the-Internet streaming is not so readily available, since the task to stream one or more video sources over the Internet to one or more recipients requires sophisticated algorithms and architectures to achieve good performance at acceptable computing loads. The challenge for these streaming video sources lies in the heterogeneous nature of Internet QoS (see figure 8): simultaneous clients have diverse connections to the Internet with different characteristics (distance, delay, throughput, jitter, errors) and

slightly different software. The answer is to individually tailor the streams to each client, but that requires powerful processors and compression algorithms able to exploit synergies between streams.

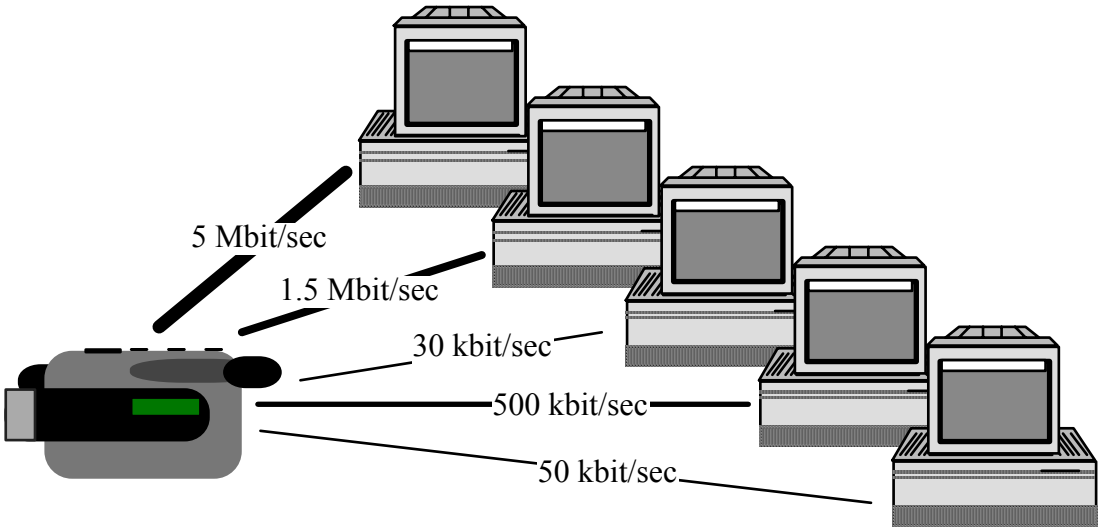

Figure 8: A Web-enabled camera distributes video to 5 receivers across lines with different QoS

WebVideo from the University of Ulm in Germany is a shareware tool for Windows PC with one or more cameras (typically PCI-bus or USB) that uses the CESC architecture to exploit synergies between the streams. The name Component Encoding - Stream Construction (CESC) already hints at the approach: one central part of the software computes components for all the streams and deposits that in a component pool. For each connected client a stream constructor selects components from the pool depending on coding requirements and the connection-specific QoS (see figure 9).

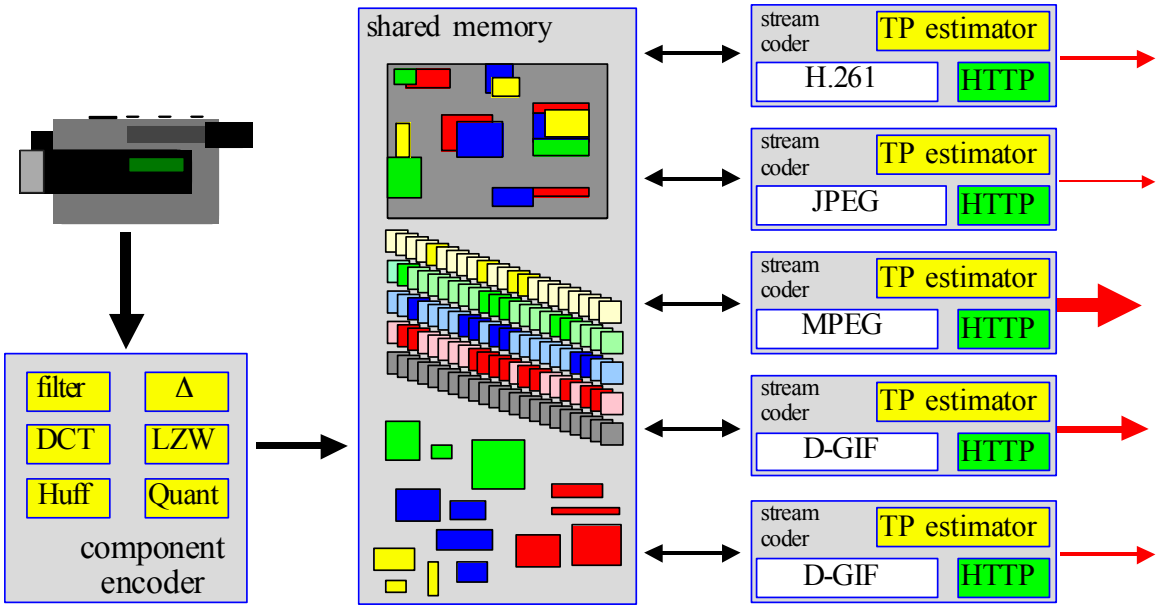

Figure 9: Component Encoding - Stream Construction prepares compressed components once and sends them several times embedded in system streams according to the actual QoS.

NuSpectra SiteCam [SiteCam] is a commercial software system to stream video simultaneously to multiple clients and remotely operate cameras. It uses several camera models with built-in pan/tilt/zoom (PTZ, see below). A nice interface enables the remote user to use the camera control. The server manages the operation rights if there are multiple clients connected. NuSpectra offers several server packages with different capabilities such as number of clients, camera control, or decoder support.

Cameras with Internet Interface are offered by several commercial companies. Many of them repackage systems from Axis, JVC, or Canon and offer services around them. Internet ready cameras are typically expensive. **Axis** produces cameras with Ethernet interfaces, which can be directly connected to the Internet. A simple Web interface allows for configuration of the camera [Axis].

Canon offers the VB101 control system which allows sharing of the Canon PTZ-cameras (e.g. VB-C10, VC-C4) [Canon]. The unit is basically a computer with Internet interface and protocols. It can be managed over the network. The VC-C4 camera (see Canon Website) has a serial interfaces to control the PTZ functions and standard video interfaces (composite, S-video). A video digitizer is needed to obtain digital video.

The **JVC** VN-C30U is a complete WebCam in a camera enclosure (see JVC Web page). It has a 10BaseT/100BaseTX interface and a built-in Webserver. The digital video performance is, however, mediocre compared to consumer DV cameras: MPEG-1 at 30 fps only with a resolution of 320x240 pixels. The higher resolution of 640*480 pixels produces 3 frames of M-JPEG per second.

More than 50 consumer WebCams are now available through mail order and retail stores. They are typically connected to the PC and WebCam software (often included with the hardware, freeware, shareware, WebVideo, SiteCam, etc.) through the parallel port (old and slow), USB, or Firewire). The decision which camera to use depends on the availability of drivers (limited for Linux and MacOS), the resolution, quality, price, and mechanical properties. The Philips ToUCam Pro for example can be mounted to oculars of telescopes and microscopes with relative ease and offers a resolution of 1280*960 pixels for still pictures.

Consumer DV cameras present the third option. Most models are now equipped with a FireWire interface (IEEE 1394, also referred to as I-Link and other proprietary names). FireWire is fast enough to transmit the DV-format (compressed at 28.77 Mbit/s). These cameras produce excellent quality video in the standard ITU-R 601 format (broadcast TV resolution).

3.2.4.4 Detecting instruments and components

During the configuration of a distributed, remotely operated experiment, the components of the system have to be 'discovered' in terms of network address, supported protocols, protocol options, and other networking oriented parameters. Three major standards in this field are relevant enough to mention in this overview:

Zeroconf is an IETF (Internet Engineering Task Force) working group creating Internet standards for attaching computers and "smart" devices (in the view of the working group members mostly PDAs, mobile phones, digital cameras, MP3-players, etc.) to the Internet

and with each other without the need for human interaction and network management. Although technically interesting, this Address Autoconfiguration, currently (in 2002) in the draft phase of IETF standardization, is beyond the scope of this report.

Very relevant however is another area of work in Zeroconf, Service Discovery, i.e. how devices dynamically detect services in any given area of the Internet. The service location protocol SLP [RFC2608] uses an administration area broadcast (typically a campus network), so that the Internet is not overloaded with the broadcast traffic. The broadcast contains a description of the requested service. Devices willing to act as such a server will then respond to the source of the broadcast. The second relevant IETF mechanism for resource discovery is the SRV resource record [RFC2782] in the database of a domain name server.

Universal Plug and Play (**UPnP**) [upnp.org] is a service discovery and configuration framework primarily for the Windows Operating System family. It defines a number of TCP/IP based protocols. SSDP (Simple Service Discovery Protocol) is used by devices to announce a service on a network over http in unicast or multicast modes. Such an announcement contains a URI for the device and a URL for the XML description of the service. Queries are sent out in a similar fashion containing a URI that is matched by the devices towards their own services. There is also the option of using a directory service. Similar to the Zeroconf approach, UPnP does not include a mechanism to invoke the service, that is left to the communication between service: and user. The XML description can however be used to define standard mechanisms for accessing the service.

JINI [Jini] from Sun Microsystems as a Java API is based on a yellow-pages paradigm: a service provider (a device providing one or more services) registers the service with a lookup server. Such a lookup server should be well known to service providers and possible clients (an alternative is to use a broadcast to discover lookup servers). The registration includes a set of tupels (name-value pairs) with service properties. The client specifies the desired service in a query to the lookup server, which tries to match the request with registered services and communicates the answer back to the client.

The next step is the most interesting part of Jini, which takes full advantage of the portable and transportable code features of Java: clients (Java applications) can load Java classes, i.e. program components, to use the service (typically implemented in Java). These classes are part of the above registration and they serve as RMI-stubs (RMI, Remote Method Invocation, the Java-RPC) to send messages to objects in the server.

Jini has some significance beyond service discovery, reaching into the area of teleprogramming. The disadvantage is however the necessity for custom programming for all devices. A user interface, for example, has to be programmed in Java.

3.3 Device independent control software

This is the area of Virtual Laboratory research and development, where the most work needs to be done. Based on abstract equipment description we have to develop standard software to interface to many different instruments. Such an effort is based on:

- A standard hardware interface for the transducer (e.g. IEEE P1451.2)

- Affordable P1451.2 adapters to Ethernet with basic IP protocols (IP, TCP, UDP, ARP, ZeroConf)
- A standard API for the instrument (register addresses and semantics)
- A set of classes to communicate with the instrument
- An electronic datasheet of the instrument (e.g. XML, XSIL)

Based on these components ISAPI filters and Apache modules can be developed, which talk to many instruments in a standardized way (see figure 10). These Webserver modules dynamically generate user-interface pages for an instrument as soon as a Web browser requests the URL of the instrument. This page will be generated from the XML electronic datasheet. Instrument data is fetched from the instrument with XML-RPCs or the Java classes of XSIL. The returned values are embedded into the Web page that is then sent to the user. JavaScript elements or Client-Pull can be embedded into the page to request regular updates of the page from the server.

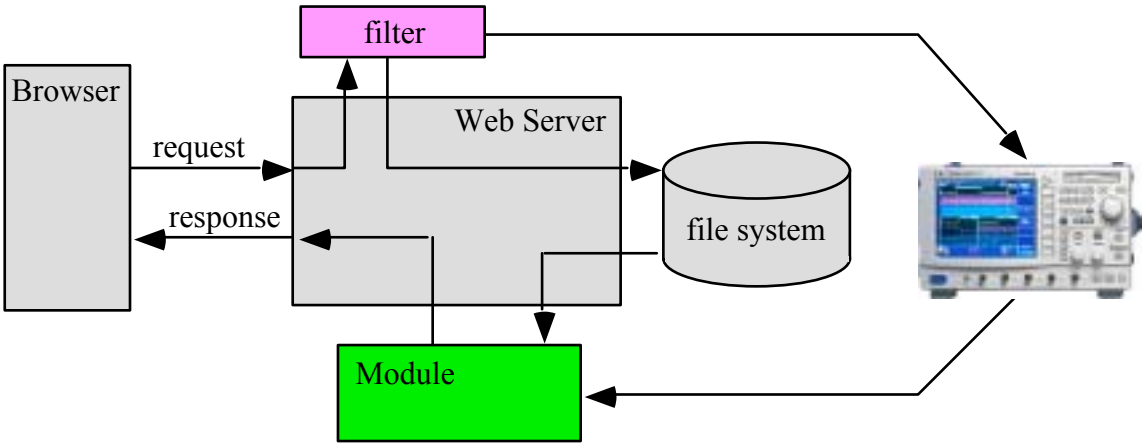

Figure 10: Webserver module as link between browser and instrument

Another implementation option for this Web-instrument gateway software is php, whereby the functions to talk to the instrument would be based on the classes mentioned above or simply XML-RPCs (php already has functions to parse XML and send XML-RPCs).

3.3.1 User Interfaces

As opposed to standard computer software, which is mostly used by non-experts, virtual laboratories are used by scientists, experts in their field. 'Ease of use', which so often determines the design of a user interface, is of minor importance in P2E communication. Designers of P2E should focus on 'Expert interfaces' to help the researcher with flexibility, fine control, and accurate data.

4 Examples

The **Internet Model RailRoad (IMRR)** is a real HO-scale model railroad, that can be controlled over the Internet with an ordinary Web browser. The moving trains can be observed with WebCams.

The basic architecture of the system can be seen in figure 11. It consists of a PC with special software for a vendor specific digital command control system (from Maerklin/Motorola, MM-DCC). The system could be adapted to work with the NMRA-DCC standard. This PC also collects sensor input from the track, which is used by a heuristic algorithm to detect the position of the train. (Although this sounds rather unreliable, this algorithm has proven very reliable combined with a few fault detection rules.)

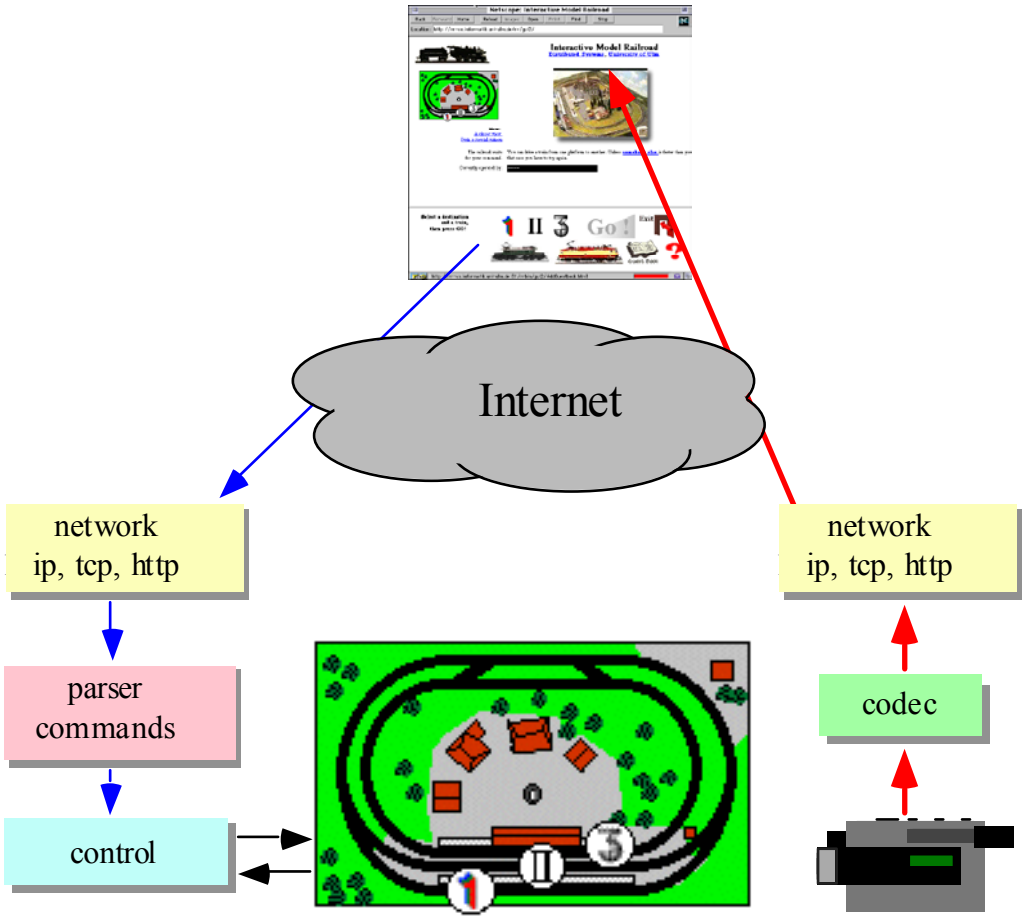

Figure 11: IMRR architecture

The equipment control components receive simple commands from a cgi-script in a separate Webserver. The fundamental command consists of a character followed by a number: train X should be moved to track n. After a conflict detection phase the control software translates the command into MM-DCC-commands to set the switch-tracks and starts the locomotives (each locomotive has an individual address set on a decoder in the locomotive). The control software then polls all the feedback contacts on the layout to detect the arrival of the moving train. As soon as the locomotive activates the contact on the destination track, it is stopped. After that the new position of the locomotive is sent to the cgi-application in the Webserver. Additional commands are used to initialize the model railroad (find the position of the locomotives on the track and move them to their starting position) and to convey the initial setting to the Webserver.

The feedback from the model railroad to the user is only through live video powered by the WebVideo system [WebVideo]. The user interface is on a relatively simple Web page, where clicks on links select states in a page tree, whose leaf nodes contain commands (figure 12). There are of course many other options for the implementation, but this particular option made the dynamic configuration of the user interface with the active locomotives very simple.

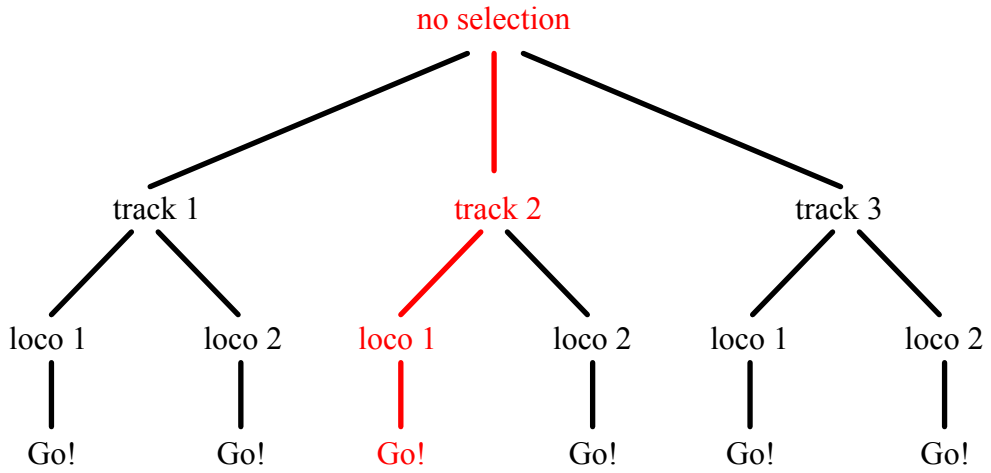

Figure 12: IMRR user interface: the commands are embedded in a page-tree

The original IMRR was located at the Distributed Systems Department of the University of Ulm in Germany. It has moved to the Computer Science Department of the Technical University Freiberg, Germany. In Freiberg the layout is now a lot bigger allowing for more active trains and more creative paths: http://www.informatik.tu-freiberg.de/bahn. Several student projects are active to explore different control schemes and new control system architectures.

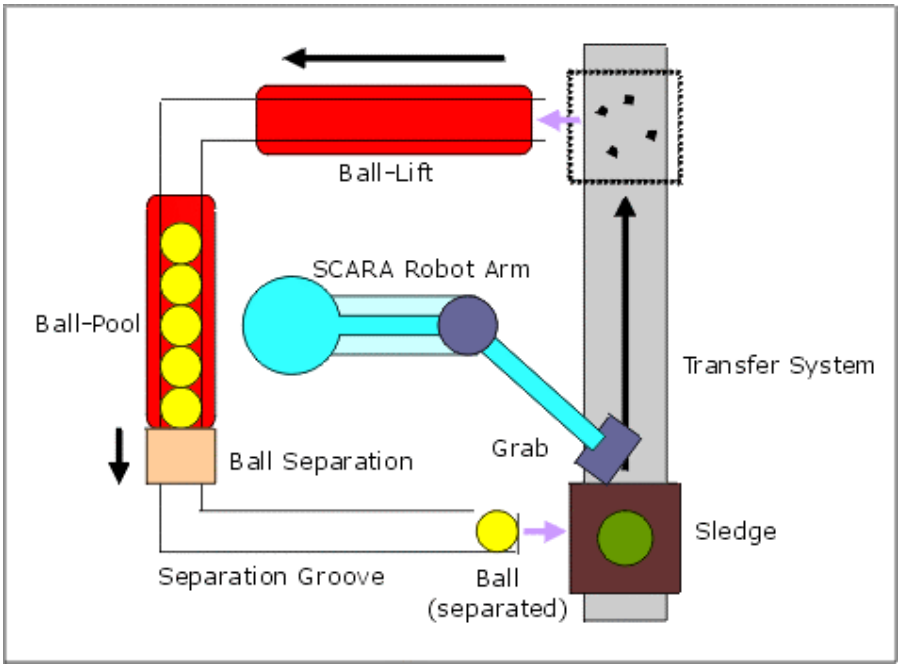

Figure 13: Teleprogramming experiment at the VVL Tübingen/Reutlingen.

In the framework of a virtual university project, the **VVL** group at the Universities of Tübingen and Reutlingen [Bühler] has created several online teaching experiments for the teleprogramming paradigm. Students write Java programs which are then downloaded into real machines with Java Virtual Machines. These applets control the hardware of the machine such as valves in a pneumatic machine (see figure 13). Camera feedback allows the students to check the function of their control programs. The experiments have been created for their educational value and all the software is custom written for the purpose. Some of the experiments are open for public Internet access [VVL].

The Tele-Presence Microscopy Project (**TPM**) at the Argonne National Laboratories (ANL) is an impressive remote controlled microscopy laboratory put together by Nestor Zaluzec [Zaluzec]. There are not only several instruments to control, an abundance of WebCams (which can be teleoperated themselves) allows the user to view the experiments, the instruments, and the whole laboratory. Due to the media rich pages, viewing and operating the experiments is best on high speed Internet connections (ADSL or better). Teleconferences are used to conduct experiments with students from schools and universities and other researchers.

NIST IEEE P1451 Demonstration [NISTDemo]

Although this demonstration is more geared to control manufacturing processes, architecture and technology could be used as a starting point for virtual laboratories scenarios. Figure 14 from the National Institute of Standards and Technology (NIST) Website gives an overview of the setup.

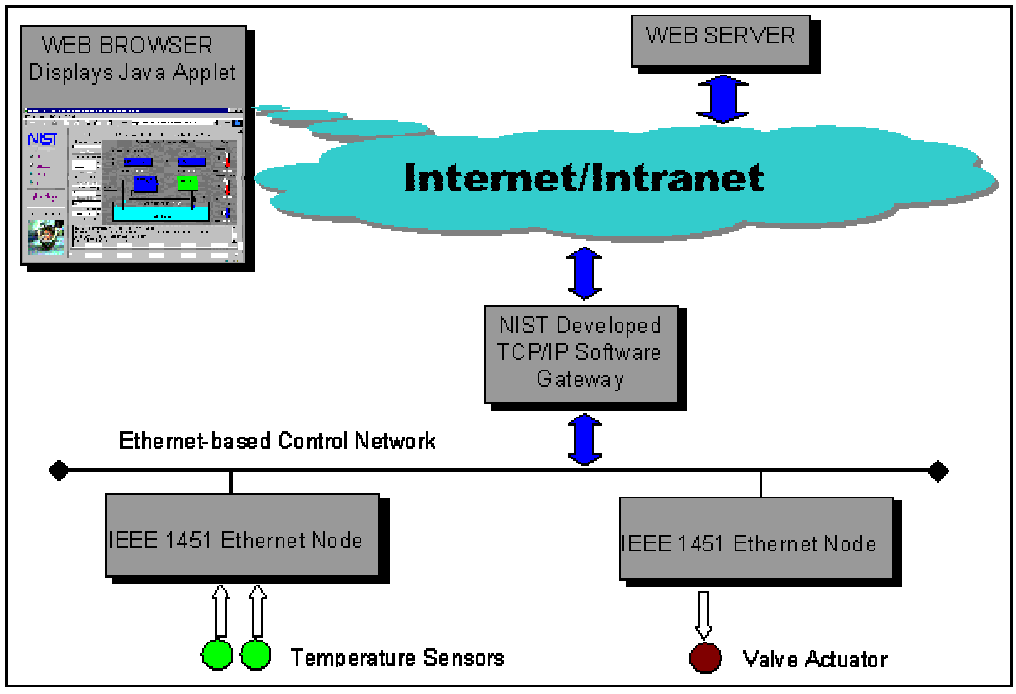

Figure 14: The NIST remote process control demonstration based on P1451 smart transducers

We quote the Website to describe the demonstration: *This demonstration uses Java and the Internet to remotely monitor and control IEEE 1451.2 networked smart transducers*

in a Manufacturing setting. A High-speed Makino Machining Center at NIST requires consistent coolant temperatures during the machining processes to reduce the amount of tool wear and to produce more exacting parts.

An application of the IEEE 1451.2 Smart Transducer Interface Module (STIM) and the IEEE P1451.1 Network Capable Application Processor (NCAP) to the high-speed machining process has been developed and deployed across the Internet. This demonstration consists of a Java Applet that is capable of monitoring and controlling the coolant temperature control loop by an operator. Live in-process IEEE 1451.2 networked smart transducers located at the NIST facility are used to highlight IEEE 1451.2. Live Video is used to enhance the monitoring capabilities of the remote demonstration [Schneemann].

A remote controlled robot [**Telerobot**] at the university of Western Australia has a WWW-based user interface for all 6 degrees of freedom offered by the robot. Feedback is again provided with WebCams, several of them in this case [Say98]. Telelabs is a new project at the same laboratory to let students conduct mechatronics experiments remotely [Telelabs].

References

Axis	http://www.axis.com
bipm	http://www.bipm.fr/enus/3_SI/si.html
Birkenmaier	Rainer Birkenmaier: Architektur und Implementierung eines schlanken WebDevices; Diplomarbeit Universität Ulm; 1998.
Bühler	D. Bühler, W. Küchlin, G. Gruhler, G. Nusser: The Virtual Automation Lab - Web-based Teaching of Automation Engineering Concepts: Proceedings of the 7th Annual IEEE International Conference on the Engineering of Computer Based Systems (ECBS), Edinburgh April 2000
Canon	http://www.canondv.com/vb101/
Fluckiger	F. Fluckiger, Networked Multimedia Prentice Hall. 1994
Funda	J. Funda, T. Lindsay, R. Paul: Teleprogramming: Toward delay-invariant remote manipulation. Presence, 1(1): 29-44, 1992.
Halsall	F. Halsall: Multimedia Communications: Applications, Networks, Protocols, and Standards; Addison Wesley, 2000;
Jini	http://www.jini.org/about/technology.html
Kidd	Eric Kidd: XML-RPC HOWTO; http://xmlrpc-c.sourceforge.net/xmlrpc-howto/xmlrpc-howto.html
Lee	Kang Lee: A Synopsis of the IEEE P1451- Standards for Smart Transducer Communication; National Institute of Standards and Technology.

MSXML	Microsoft Developer Network: A Guide to XML and Its Technologies; http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnxml/html/xmlguide.asp
NIST1451	http://ieee1451.nist.gov/
NISTDemo	http://motion.aptd.nist.gov/P1451/ISADemo.htm
Phpfunctions	http://www.php.net/manual/en/funcref.php
Phpman	http://www.php.net/manual/en/tutorial.php
RFC 2608	http://www.faqs.org/rfcs/rfc2608.html
RFC 2782	http://www.faqs.org/rfcs/rfc2782.html
SiteCam	http://www.nuspectra.com/
Sayers	Craig Sayers: Remote Control Robotics; Springer, 1999.
Schneemann	R. Schneemann: NIST Demonstration Overview; http://motion.aptd.nist.gov/P1451/Docs/info.htm
Telelabs	http://www.mech.uwa.edu.au/jpt/tele/Telelabs.html
Telerobot	http://telerobot.mech.uwa.edu.au
Timbuktu	http://www.netopia.com/en-us/software/products/tb2/index.html
upnp.org	http://www.upnp.org/
VVL	http://www.vvl.de/VVLenglish/index.html
WebVideo	K. Froitzheim, H. Wolf: WebVideo - a Tool for WWW-based Teleoperation, IEEE International Symposium on Industrial Electronics, Guimaraes, Portugal, July 1997.
Williams	Roy Williams: XSIL: Java/XML for Scientific Data; http://www.cacr.caltech.edu/projects/xsil/xsil_spec.pdf
Woods	Stan P. Woods et al.: IEEE-P1451.2 Smart Transducer Interface Module; http://ieee1451.nist.gov/senoct11-2col.pdf
xml.org	http://www.xml.org/xml/resources_focus_beginnerguide.shtml
xmlrpc.com	http://www.xmlrpc.com/
Zalusec	Nestor Zalusec: http://tpm.amc.anl.gov/

Glossary

ADC	Analog to Digital Converter
ADSL	Asymmetric Digital Subscriber Line, a modulation scheme for telephony subscriber lines to enable high-volume data transmission
ANL	Argonne National Laboratories

API	Application Programming Interface, a set of programming and messaging conventions to access functions in another software module or operating system component
ARP	Address Resolution Protocol to dynamically map IP addresses to physical (hardware) addresses on local area network
asp	Microsoft Active Server Page, a web technology for dynamically changing the content of Webpages and scripting
CESC	Component Encoding Stream Construction
CGI	Common Gateway Interface, an interface between Webservers and standard computer programs such these programs are loaded and executed under the control of the http server. Parameters and results can be passed.
CORBA	Common Object Request Broker Architecture, a standard to find and use objects in distributed systems.
DAC	Digital to Analog Converter
DCE	Distributed Computing Environment
DCT	Discrete Cosine Transformation
D-GIF	GIF allows for dynamic replacement of parts of the image. This can be used to implement Difference encoding.
DiffServ	Differentiated Services, a set of standards to implement QoS in the Internet
DLL	Dynamic Link Libraries
DNS	Domain Name System, a internet service translating host names to IP addresses
DTD	Document Type Definition, a mechanism to specifiy syntactic rules in SGML and XML documents
DV	Digital Video
GIF	Graphics Image Format, a popular image file format
H.261	A video compression standard from the ITU.
HCI	Human Computer Interfaces
HO-scale	A scale and gauge of model railroads
HTML	Hypertext Markup Language, a simple language to create documents for the Web
HTTP	Hypertext Transfer Protocol, fundamental protocol for the Web technology
I/O	Input/Output
IDL	Interface Decription Language, a CORBA component

IEEE	Institute of Electrical and Electronic Engineers (New York, USA)
IEEE P1451.1	Standard of Network Capable Application Processor Information Model
IEEE P1451.2	Standard of Transducer to Microprocessor Communication Protocols and Transducer Electronic Data Sheet Format
IEEE P1451.3	Standard of Network Capable Application Processor Information Model
IEEE P1451.4	Standard of Mixed-mode Communication Protocols and Transducer Electronic Data Sheet Formats
IETF	Internet Engineering Task Force, the somewhat informal standardization organization for Internet technologies
IIS	Microsofts Internet Information Server
IMRR	Internet Model RailRoad
IntServ	Integrated Services, a set of standards to implement QoS in the Internet
IP	Internet Protocol, a technology which allows packets from one IP address to another IP-adress over the Internet
ISAPI	Internet server application program interface. a programming interface on IIS
ISDN	Integrated service digital network, the digital telephony system
ITU-R 601	International Telecommunication Union – Radio 601, a video format
JPEG	A popular image file format named after it's standardization committee, the Joint Photographic Experts Group.
LZW	Lempel-Ziv-Welch, a refinement of the LZ77 algorithm for dictionary based data compression
MFC	Microsoft Foundation Classes, a set of objects to access functionality of the Windows operation system.
M-JPEG	Motion-JPEG, a non-standard application of JPEG to digital video.
MM-DCC	Motorola-Märklin – Digital Command Control, a proprietary scheme to digitally control model railroad components over the track.
mod_php	Apache modul for PHP
MP3	MPEG Audio layer 3, an audio compression scheme and stream format
MPEG	A standard for moving images, named after it's standardization committee, the Motion Pictures Experts Group.
NCAP	Network Capable Application Processor
NIST	National Institute of Standards and Technology
NMRA-DCC	National Model Railroad Association – Digital Command Control, a standard to digitally control model railroad components over the track.
P2E	Person to Equipment

PCI	A bus to connect peripheral devices to the processor of a computer
PDA	Personal Digital Assistant, a handheld computer
PPP	Point-to-Point Protocol, a protocol to transports packets over modem lines
PTZ	Pan Tilt Zoom, movement functions in a fully motorized video camera
QoS	Quality of Service, a set of attributes that describes a service, e.g. throughput, jitter, error probability.
RFC	Request For Comments, a series of publications of networking technical documents
RMI	Remote Method Invocation, an object-oriented remote procedure call mechanism for Java
RPC	Remote Procedure Call, client/server model for distributed computing. network.
RPM	Revolutions per minute
SGML	Standard generalized markup language
SI system	Système International d'Unités [bipm]
SLIP	Serial Line Internet Protocol, a protocol to transports packets on point-to-point communication lines such as modems
SLP	Service Location Protocol
SQL	Structured Query Language, a standard language for creating, updating and, querying relational database management systems.
SRV	Server Resource Record
SSDP	Simple Service Discovery Protocol
STIM	Smart Transducer Interface Module
S-video	a component video signal in which the luminance (Y) and chrominance (C) information use separate lines as opposed to composite video.
TCP	Transmission Control Protocol, a standard for transmitting data on the Internet
TEDS	Transducer Electronic Data Sheet
TP estimator	Throughput estimator
TPM	Tele-Presence Microscopy Project
UPnP	Universal Plug and Play
URI	Uniform resource identifier, a identifier or address for a resource in the Internet
URL	Uniform resource locator, special type of URI

USB	Universal serial bus, a hardware interface supporting different desktop computer peripherals on one port
VCR	Video recorder
VL	Virtual Laboratories.
VR	Virtual Reality
VVL	Virtual University Project (Verbund Virtuelles Lernen)
W3C	World Wide Web Consortium
WebIR	WWW Infrared Remote control
WWW	World Wide Web
XDR	the eXternal Data Representation language
XML	eXtensible Markup Language, a standard format for data on the internet
XSIL	Extensible Scientific Interchange Language, an XML-based extensible, hierarchical, transport language for scientific data objects
XSLT	eXtensible Stylesheet Language Transformations; a programming language to transform XML documents