

VIRTUAL INSTRUMENTATION

ŽELJKO OBRENOVIĆ
 DUŠAN STARČEVIĆ
 EMIL JOVANOVIĆ
 University of Alabama in
 Huntsville
 Huntsville, AL

1. INTRODUCTION

Virtual instrumentation is an interdisciplinary field that merges sensing, hardware, and software technologies to create flexible and sophisticated instruments for control and monitoring. Several definitions of a virtual instrument are available in the open literature. Santori defines a virtual instrument as “an instrument whose general function and capabilities are determined in software” (1). Goldberg states that “a virtual instrument is composed of some specialized subunits, some general-purpose computers, some software, and a little know-how” (2). Although informal, these definitions capture the basic idea of virtual instrumentation and virtual concepts in general, provided with sufficient resources, “any computer can simulate any other if we simply load it with software simulating the other computer” (3). This universality introduces one basic property of a virtual instrument: its ability to change form through software, enabling a user to modify its function at will to suit a wide range of applications. The concept of *virtual instrumentation* was born in the late 1970s, when microprocessor technology enabled a machine’s function to be more easily changed by changing its software (1). This flexibility is possible as the capabilities of virtual instruments depend very little on dedicated hardware — commonly, only the application-specific signal conditioning module and the analog-to-digital converter used as an interface to the external world. Therefore, simple use of computers or specialized onboard processors in instrument control and data acquisition cannot be defined as virtual instrumentation.

An increasing number of biomedical applications use virtual instrumentation to improve insights into the underlying nature of complex phenomena and reduce costs of medical equipment and procedures (4). Although many general virtual instrumentation concepts may be directly used in biomedical measurements, the measurements in the medical field are peculiar as “they deal with a terribly complex object — *the patient* — and are performed and managed by another terribly complex instrument — *the physician*” (5).

In this article, we describe basic concepts of virtual instrumentation as well as biomedical applications of virtual instrumentation. In the second section, we give a brief history of virtual instrumentation. The architecture of a virtual instrument is described in the third section. In the fourth section, we describe the organization of distributed virtual instrumentation. After that, we present contemporary virtual instrument development tools. Finally, we present some biomedical applications of virtual instrumentation.

2. A BRIEF HISTORY OF VIRTUAL INSTRUMENTATION

A history of virtual instrumentation is characterized by a continuous increase of flexibility and scalability of measurement equipment. Starting from manually controlled vendor-defined electrical instruments, the instrumentation field has made a great progress toward contemporary computer-controlled, user-defined, sophisticated measuring equipment. Instrumentation development can be traced through the following phases:

- Analog measurement devices
- Data acquisition and processing devices
- Digital processing based on a general-purpose computing platform
- Distributed virtual instrumentation

The first phase is represented by early “pure” analog measurement devices, such as oscilloscopes or electroencephalography (EEG) recording systems. These devices were completely closed dedicated systems, which included power supplies, sensors, translators, and displays (6). They required manual settings, presenting results on various counters, gauges, cardiac resynchronization therapy (CRT) displays, or paper. Additional use of data was not part of the instrument package, and operators had to physically copy data to a paper notebook or a data sheet. Performing complex or automated test procedures was complicated or impossible, as everything had to be set manually.

The second phase started in the 1950s as a result of demands from the industrial control field. Instruments incorporated rudiment control systems, with relays, rate detectors, and integrators. That led to creation of proportional-integral-derivative (PID) control systems, which allowed greater flexibility of test procedures and automation of some phases of the measuring process (2). Instruments started to digitalize measured signals, which allowed digital processing of data and introduction of more complex control or analytical decisions. However, real-time digital processing requirements were too high for any but an onboard special-purpose computer or digital signal processor (DSP). The instruments were still stand-alone, vendor-defined boxes.

In the third phase, measuring instruments became computer based and begun to include interfaces that enabled communication between the instrument and the computer. This relationship started with the general-purpose interface bus (GPIB) originated in the 1960s by Hewlett-Packard (HP), then called HPIB, for the purpose of instrument control by HP computers. Initially, computers were primarily used as offline instruments. They further processed the data from recorded measurements stored on a disk or tape (7).

As the speed and capabilities of general-purpose computers, advanced exponentially general-purpose computers became fast enough for complex real-time measurements. It soon became possible to adapt conventional high-speed computers to the online applications required in real-time measurement and control. New

2 VIRTUAL INSTRUMENTATION

general-purpose computers from most manufactures incorporated all hardware and much of the general software required by the instruments for their specific purposes. The main advantages of commonplace personal computers are the low price driven by the large market, availability, and standardization.

Although the computers' performance soon became high enough, computers were still not easily used by ordinary experimentalists. Nearly all early instrument control programs were written in BASIC, because it had been the dominant language used with dedicated instrument controllers. Therefore, it required engineers and other users to become programmers, which made it hard for them to exploit the potential that computerized instrumentation could bring. Therefore, an important milestone in the history of virtual instrumentation was the introduction of LabVIEW 1.0 on a PC platform by National Instruments in 1986 (1). LabVIEW introduced graphical user interfaces and visual programming into computerized instrumentation, joining the simplicity of user interface operations with the increased capabilities of computers. Today, the PC is the platform on which most measurements are made, and the graphical user interface has made measurements more user-friendly.

As a result, virtual instrumentation made possible a significant decrease in the price of instruments. As virtual instruments depend very little on dedicated hardware, customers could use their computers, and instrument manufacturers could supply only what the user could not get in the general market.

The fourth phase became feasible with the development of local and global networks of general-purpose computers. Because most instruments were already computerized, advances in telecommunications and network technologies made possible physical distribution of virtual instrument components into telemedical systems to provide medical information and services at a distance. The possible infrastructure for distributed virtual instrumentation includes the Internet, private networks, and

cellular networks, where the interface between the components can be balanced for price and performance.

3. VIRTUAL INSTRUMENT ARCHITECTURE

A virtual instrument is composed of the following blocks:

- Sensor module
- Sensor interface
- Medical information systems interface
- Processing module
- Database interface
- User interface

Figure 1 shows the general architecture of a virtual instrument.

The sensor module detects physical signals and transforms them into an electrical form, conditions the signal, and transforms it into a digital form for additional manipulation. Through a sensor interface, the sensor module communicates with a computer. Once the data are in a digital form on a computer, they can be processed, mixed, compared, and otherwise manipulated or stored in a database. Then, the data can be displayed or converted back to an analog form for additional process control. Biomedical virtual instruments are often integrated with some other medical information systems such as hospital information systems. In this way, the configuration settings and the data measured may be stored and associated with patient records.

In the following sections, we describe in more detail each virtual instruments module.

3.1. Sensor Module

The sensor module performs signal conditioning and transforms the signal into a digital form for additional manipulation. The sensor module interfaces a virtual instrument to the external, mostly analog world, transforming measured signals into a computer readable form.

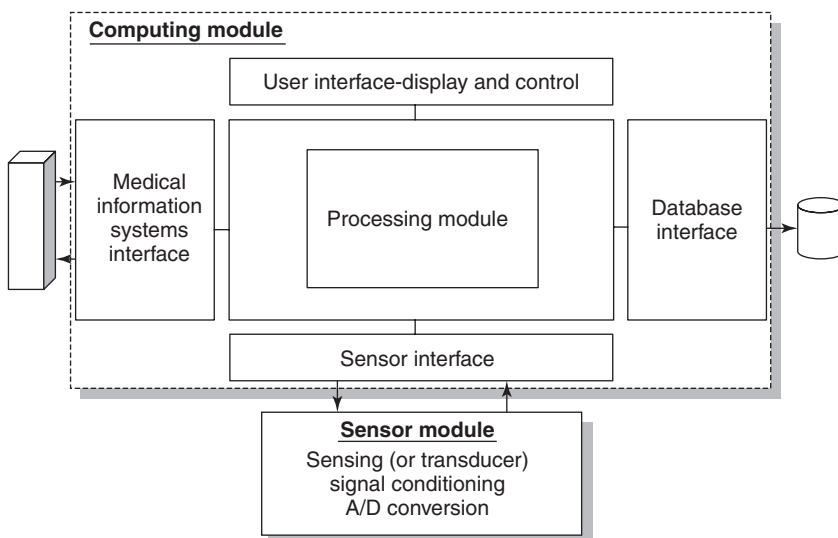


Figure 1. Architecture of a virtual instrument

Table 1. Commonly Used Biomedical Data Formats.

Group	Physiological Signal
Electrical signals (requires only amplification)	Electromyograph (EMG)
	Electrocardiograph (ECG)
	Electroencephalograph (EEG)
	Electrooculograph (EOG)
Non-electrical signals (require a transducer to change the information to an electrical signal)	Skin conductivity (Galvanic Skin Response - GSR)
	Respiratory rate
	Blood pressure
	Peripheral body temperature

Table 1 summarizes some often used human physiological signals (8).

A sensor module principally consists of three main parts:

- Sensor
- Signal conditioning part
- A/D converter.

The sensor detects physical signals from the environment. If the parameter being measured is not electrical, the sensor must include a transducer to convert the information to an electrical signal, for example, when measuring blood pressure. According to their position, biomedical sensors can be classified as follows:

- *Implanted sensors*, where the sensor is located inside the user's body, for example, intracranial stimulation.
- *On-the-body sensors*, which are the most commonly used biomedical sensors. Some of those sensors, such as EEG or electrocardiography (ECG) electrodes, require additional gel to decrease contact resistance.
- *Noncontact sensors*, such as optical sensors and cameras that do not require any physical contact with the object being measured.

The signal-conditioning module performs (usually analog) signal conditioning before A/D conversion. This module typically does the amplification, transducer excitation, linearization, isolation, or filtering of detected signals.

The A/D converter changes the detected and conditioned voltage into a digital value (9). The converter is defined by its resolution and sampling frequency. The converted data must be precisely time-stamped to allow later sophisticated analyses (1).

Although most biomedical sensors are specialized in processing of certain signals, it is possible to use generic measurement components, such as data acquisition (DAQ), or image acquisition (IMAQ) boards, which may be applied to a broader class of signals. Creating generic measuring boards, and incorporating the most important components of different sensors into one unit, it is possible

to perform functions of various medical instruments on the same computer (4).

3.2. Sensor Interface

Many interfaces can be used for communication between sensor modules and the computer. According to the type of connection, sensor interfaces can be classified as *wired* and *wireless*.

- *Wired Interfaces* are usually standard parallel interfaces, such as GPIB, small computer systems interface (SCSI), system buses (PCI eXtension for Instrumentation—PXI or VME extensions for instrumentation—VXI), or serial buses (RS232 or USB interfaces) (10).
- *Wireless Interfaces* are increasingly used because of convenience. Typical interfaces include the IEEE 802.11 family of standards, Bluetooth, or GPRS/GSM interface (11). Wireless communication is especially important for implanted sensors in which the cable connection is impractical or not possible (12). In addition, standards, such as Bluetooth, define a self-identification protocol, which allow the network to configure dynamically and describe itself. In this way, it is possible to reduce installation cost and create plug-and-play like networks of sensors. Device miniaturization allowed development of personal area networks (PANs) of intelligent sensors (13,14).

Communication with medical devices is also standardized with the IEEE 1073 family of standards (15,16). This interface is intended to be highly robust in an environment where devices are frequently connected to and disconnected from the network.

3.3. Processing Module

Integration of the general-purpose microprocessors/microcontrollers allowed flexible implementation of sophisticated processing functions. As the functionality of a virtual instrument depends very little on dedicated hardware, which principally does not perform any complex processing, functionality and appearance of the virtual instrument may be completely changed using different processing functions.

Broadly speaking, processing functions used in virtual instrumentation may be classified as *analytic processing* and *artificial intelligence techniques*.

3.3.1. Analytic Processing. Analytic functions define clear functional relations among input parameters. Some common analyses used in virtual instrumentation include spectral analysis, filtering, windowing, transforms, peak detection, or curve fitting (17). Virtual instruments often use various statistics function, such as random assignment and biostatistical analyses (18). Most functions can nowadays be performed in real time.

3.3.2. Artificial Intelligence (AI) Techniques. AI technologies could be used to enhance and improve efficiency, capabilities, and features of instrumentation in application

4 VIRTUAL INSTRUMENTATION

areas related to measurement, system identification, and control (19,20). These techniques exploit the advanced computational capabilities of modern computing systems to manipulate sampled input signals and extract the desired measurements.

AI technologies, such as neural networks, fuzzy logic, and expert systems, were applied in various applications, including sensor fusion to high-level sensors, system identification, prediction, system control, complex measurement procedures, calibration, and instrument fault detection and isolation (21). Various nonlinear signal processing, including fuzzy logic and neural networks, are also common tools in analysis of biomedical signals (22,23).

Using AI, it is even possible to add medical intelligence to ordinary user interface devices. For example, several AI techniques, such as pattern recognition and machine learning, were used in a software-based visual-field testing system (24). Visual-field testing, called perimetry, can help in early detection of various eye diseases such as optic neuritis or glaucoma. A proposed self-screening system uses software-controlled perimetry that operates on PCs, where crucial positions in the visual field of subjects relate to specific locations on the screen. The test screen presents several objects of the same type, where one of them, the stimulus, is moving, and subjects respond to stimuli by clicking the mouse. A computational method for identifying measurement noise uses Kohonen's self-organizing maps with two layers of nodes. Therefore, AI techniques such as intelligent user interface, software-based perimetry, and pattern discovery method replace medical operators that otherwise have to constantly monitor the subject during the test and even perform initial screening without doctors on a general-purpose computer.

3.4. Database Interface

Computerized instrumentation allows measured data to be stored for offline processing or to keep measurement records as a part of the patient record (25). The database can also store configuration settings. Several currently available database technologies can be used for this purpose (Table 2).

Simple usage of a file system interface leads to the creation of many proprietary formats, so the interoperability may be a problem. The eXtensible Markup Language (XML) may be used to solve the interoperability problem by providing universal syntax (26). The XML is a standard for describing document structure and content (27). It organizes data using markup tags, creating self-describing documents, as tags describe the information it contains. Contemporary database management systems, such as the Structured Query Language (SQL) Server and Oracle, support XML import and export of data.

Many virtual instruments use (DBMSs). They provide efficient management of data and standardized insertion, update, deletion, and selection. Most of these DBMSs provided an SQL interface, enabling transparent execution of the same programs over databases from different vendors. Virtual instruments exploit these DBMSs by using some

Table 2. The Most Frequently Used Contemporary Databases Interfaces

Data System	Description
File system	Random writing and reading of files
eXtensible Markup Language (XML)	Standardized markup files
Open Database Connectivity (ODBC)	SQL based interface for relation databases
Java Database Connectivity (JDBC)	Java programs' SQL based object-oriented interface for various databases
Active X Data Objects (ADO)	Windows programs' object-based interface for various data sources including relational databases and XML files
Data Access Objects (DAO)	Windows programs' object-based interface for relation databases

programming interfaces, such as ODBC, JDBC, ADO, and DAO (28).

3.5. Medical Information System Interface

Virtual instruments are increasingly integrated with other medical information systems, such as hospital information systems. Consequently, they can be used to create executive dashboards, assisting decision support, real-time alerts, and predictive warnings (29). Some virtual interfaces toolkits, such as LabView, provide mechanisms for customized components, such as ActiveX objects (28), which allow communication with other information system, hiding the details of the communication from virtual interface code.

In Web-based telemedical applications, this integration is usually implemented with unified resource locators (URLs) as Fig. 2 shows. Each virtual instrument is identified with its URL, receiving configuration settings via parameters. The virtual instrument then can store the results of the processing into a database identified with its URL (30).

In addition to described integration mechanisms, there are standards for communications among medical applications. For example, the Object Management Group (OMG) Healthcare DTF (<http://healthcare.omg.org/>) is defining standards and interfaces for health-care objects, such as CORBAMED standard, to develop interoperability technologies for the global health-care community (31). Although these standards are still not widely used, they have the potential to ensure interoperability among virtual instruments and medical information systems on various platforms.

3.6. Presentation and Control

User interface affects efficiency and precision of a virtual instrument and facilitates result interpretation. Because a computer's user interfaces are much more easily shaped and changed than a conventional instrument's user interfaces, it is possible to employ more presentation effects and customize the interface for each user. According to pre-

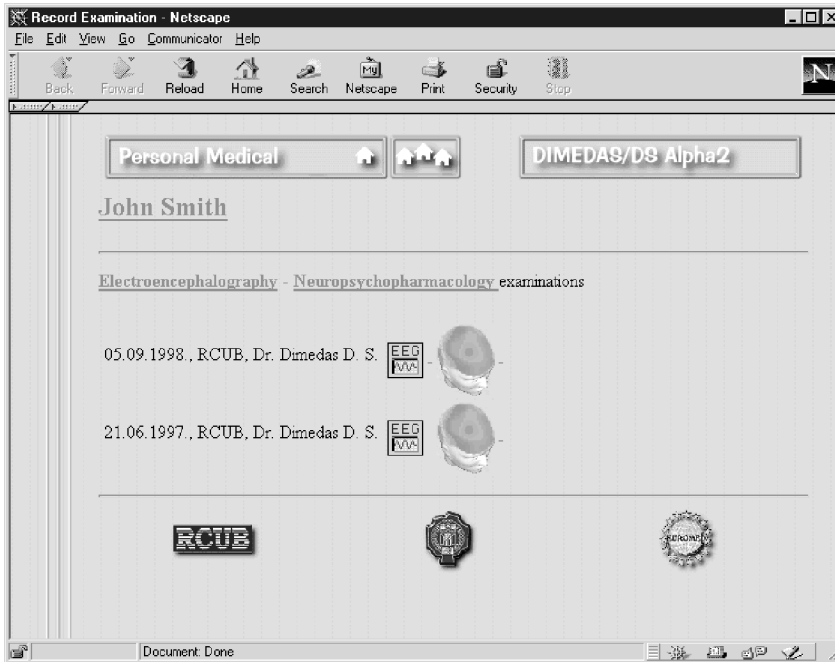


Figure 2. EEG virtual medical devices (VMDs) in DIMEDAS information system; two VMDs are associated with every EEG recording.

resentation and interaction capabilities, we can classify interfaces used in virtual instrumentation into four groups:

- Terminal user interfaces
- Graphical user interfaces
- Multimodal user interfaces
- Virtual and augmented reality interfaces

3.6.1. Terminal User Interfaces. The first programs for instrumentation control and data acquisition had character-oriented terminal user interfaces, which was necessary as earlier general-purpose computers could not present complex graphics. As terminal user interfaces required little of system resources, they were implemented on many platforms.

In this interface, communication between a user and a computer is purely textual. The user sends requests to the computer typing commands and receives a response in a form of textual messages. Presentation is usually done on a screen with fixed resolution, for example, 25 rows and 80 columns on an ordinary PC, where each cell presents one character from a fixed character set, such as the ASCII set. Additional effects, such as text and background color or blinking, are possible on most terminal user interfaces. Even with the limited set of characters, more sophisticated effects in the form of character graphics are possible.

Although terminal user interfaces are not any more widely use on desktop PCs, they have again become important in a wide range of new pervasive devices, such as cellular phones or low-end personal digital assistants (PDAs). As textual services, such as short message service (SMS), require small presentation and network resources, they are broadly supported and available on almost all cellular phone devices. These services may be important in distributed virtual instrumentation and for emergency alerts (32).

3.6.2. Graphical User Interfaces (GUI). Graphical user interfaces (GUIs) enabled more intuitive human–computer interaction, making virtual instrumentation more accessible (1). Simplicity of interaction and high intuitiveness of GUI operations made possible the creation of more user-friendly virtual instruments. GUIs allowed the creation of many sophisticated graphical widgets such as graphs, charts, tables, gauges, or meters, which can easily be created with many user interface tools.

Computer graphics extended the functionality of conventional medical diagnostic imaging in many ways, for example, by adding the visual tool of color. For instance, interpretation of radiographs, which are black-and-white images, requires lots of training, but with color, it is possible to highlight problems clearly (4). In addition, improvements in presentation capabilities of personal computers allowed for development of various sophisticated two- (2-D) and three-dimensional (3-D) medical imaging technologies (33).

3.6.3. Multimodal Presentation. In addition to GUIs, contemporary personal computers can present other modalities such as sonification or haptic rendering. Multimodal combinations of complementary modalities can greatly improve the perceptual quality of user interfaces (34,35).

Sonification is the second-most important presentation modality. Although still limited, acoustic user interfaces are becoming more common with basic sound hardware now available in most computing systems. When the visual field is overwhelmed, audio feedback and synchronized auditory and haptic stimuli may be useful. Efficiency of sonification, as acoustic presentation modality, depends on other presentation modalities. Especially interesting is the relationship between visualization and sonification,

6 VIRTUAL INSTRUMENTATION

which is a complex design problem, because of the nature of the cognitive information processing.

Typical applications of acoustic presentation include tactical audio and sonification. Tactical audio provides acoustic guidance through the use of audio feedback for facilitating precise and accurate positioning of an object with respect to another object. This type of feedback has valuable application in the field of computer-assisted surgery, in which multiple parallel voices provide independent channels of positional information, used as a feedback during simulation or operation (34). Sonification is effectively used in various biomedical applications as alternative presentation modality. For example, sonification of EEG sequences has been applied to detect short-time synchronization during cognitive events and perception (36). Another example is the application of an audible display of patient respiratory data for anesthesiologists in the operating room. An audible display provides information about the patient's status during surgery because anesthesiologists spend only 7% to 20% of their intraoperative time observing patient monitors (37).

Although not widely available, haptic rendering may be an important upcoming presentation modality for virtual instruments, as the physical contact between a physician and a patient is part of standard examination procedures. Commercially available haptic interfaces can relay resistance at about 1000 Hz and are used in various surgical simulations (38).

3.6.4. Virtual and Augmented Reality. Virtual environments will most likely pervade the medical practice of the future (39). Many goals of virtual reality developers actually mirror those involved in virtual instrumentation (4). Although virtual reality systems do not necessarily involve the use of virtual instrumentation, they, nonetheless, drive the development of new circumstances under which physicians will need access to data in radically different forms (38).

A combination of virtual presentation with real-world objects creates augmented reality (AR) interfaces. In contrast to virtual reality systems, normally designed to immerse the user as completely as possible within a computer-generated environment, AR extends real-world images with computer-generated elements. Visually, this is realized by electronic or optical superimposition of computer graphics with a user's view of the real world (40). For example, AR may allow a computer-generated tumor image from a magnetic resonance image recording to be superimposed on the real view of the patient during surgery.

3.7. Functional Integration

Functional integration of modules governs flexibility of a virtual instrument. The simplest, and the least flexible, way is to create a virtual instrument as a single, monolithic application with all software modules of the virtual instruments logically and physically integrated. This approach can achieve the best performance, but it makes maintenance and customization difficult. Therefore, it is more convenient to use modular organization. An object-

oriented method was identified as a natural approach in modeling and design of instruments (41,42). Each module of a virtual instrument is implemented as an object with a clearly defined interface, integrated with other objects using message interchange. A similar approach is a component-oriented approach, where, in addition to logical separation of components into objects, they are physically placed into different units to allow reuse.

Another approach, similar in its basic idea to the object-oriented approach, is a structural coupling paradigm that defines a layered approach to functional integration of sensor modules (43). This sensor model for nonconventional controllers was applied in many domains, including electrophysiological interaction systems with sensors for human physiological signals (44). In this sensor interaction model, a stream of raw data from the sensing hardware, for example, EEG data, passes through up to two levels of signal preprocessing before it is either passed to an application or presented directly to a subject. The second command layer, which is optional, allows more flexible organization of data processing and plug-and-play-like integration of complex processing mechanisms into a virtual instrument solution.

4. DISTRIBUTED VIRTUAL INSTRUMENTATION

Advances in telecommunications and network technologies made possible physical distribution of the virtual instrument components into telemedical systems to provide medical information and services at a distance (45). Distributed virtual instruments are naturally integrated into telemedical systems (46). Figure 3 illustrates the possible infrastructure for distributed virtual instrumentation, in which the interface between the components can be balanced for price and performance (2).

4.1. Medical Information System Networks and Private Networks

Medical information systems, such as hospital information systems, are usually integrated as intranets using a local area network (LAN). Historically, medical information systems were first interconnected using various private networks, starting from point-to-point

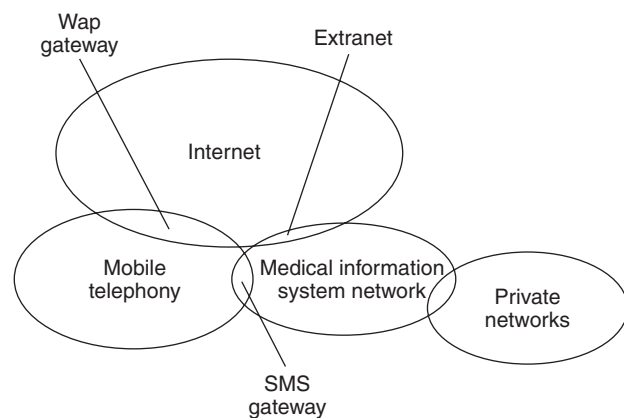


Figure 3. Infrastructure for distributed virtual instrumentation.

communication with fax and modems over analog telephone lines operating at speeds up to 56 Kbps, ISDN lines of up to 128 Kbps, T-1 lines having a capacity of 1.544 Mbps, and satellite links of 100 Mbps.

Advanced virtual instrumentation solutions could be implemented using existing local and private networks (47). For example, the Experimentalist's Virtual Acquisition Console (EVAC) project demonstrated a prototype system for using virtual environments to control remote instrumentation, which illustrated the potential of a virtual laboratory over high-speed networks (48). The system has five discrete elements: virtual visualization environment, voice command instrumentation control, instrumentation data transfer, audio collaboration link, and video collaboration link. The main objective of the EVAC project was to demonstrate a unified user environment that allows access over high-speed networks to hardware in a remote experimental laboratory. However, although private networks improve the performance, reliability, and security, they are usually expensive to develop and maintain.

4.2. The Internet

The Internet has enormous potential for distributed biomedical virtual instrumentation. Various remote devices, such as telerobots or remote experimental apparatus, can be directly controlled from the Internet (49). Many research activities explore how the Internet can be applied to medicine (31). In addition, many virtual instrumentation development tools, such as LabVIEW (50), directly support integration of virtual instruments in the Internet environment (30,51). The Web technologies make possible the creation of sophisticated client-server applications on various platforms, using interoperable technologies such as HTML, Java applets, Virtual Reality Modeling Language (52), and multimedia support (30).

Although the Internet is already enabling technology for many biomedical applications, a recent study of health-care applications in United States in relation to Internet capabilities found clear requirements for the Internet's evolutionary development (53). More serious use of the Internet in clinical applications could be achieved only if the level of service can be guaranteed, including a consistent level of bandwidth end-to-end as well as high reliability and security (45). However, the rapid progress of the Internet will probably soon enable its usage in complex real-time applications.

4.3. Cellular Networks

Various mobile devices, such as mobile phones or PDAs, are commonplace today. Moreover, the underlying telecommunication infrastructure of these devices, primarily cellular networks, provides sophisticated data services that can be exploited for distributed applications. The most common data service on cellular networks is exchange of simple textual messages. Most mobile phone devices support simple data communication using the standard SMS. Although simple, this system allows the various modes of communication for medical applications (32) such as:

- *Emergency SMS push*, which sends SMS message to physicians or a medical call center in case of medical emergency
- *SMS query*, which allows the user to ask for some simple information in medical information systems or in a monitoring system
- *Data distribution SMS push*, which periodically sends SMS messages with some monitoring data to physicians' emergency SMS push.

Wireless access protocol (WAP) is platform-independent wireless technology, which enables mobile devices to effectively access Internet content and services, as well as to communicate with each other (54). WAP manages communication by exchanging messages written in Wireless Markup Language (WML). The WAP and the Internet can support new kinds of applications, such as remote monitoring using a wireless personal monitor and cellular phone link connected on request in the case of medical emergencies (13,14). The interface allows the following modes of communications (32):

- *Emergency WAP push*, which sends WML messages to physicians or a medical call center in case of medical emergency
- *WML browsing*, which allows a participant to browse through information in medical information systems or in monitoring system
- *Data distribution WAP*, which periodically sends messages to physicians. These data could be simple text or some 2-D graphics with wireless bitmaps (WBMPs).

4.4. Distributed Integration

According to a conceptual model and abstractions they use, we can identify four approaches to distributed communication:

- Message passing systems
- Remote procedure calling (RPC) systems
- Distributed object systems
- Agent-based systems

The message passing model allows communication between programs by exchanging messages or packets over the network. It supports a variety of communication patterns, such as pier-to-pier, group, broadcast, and collective communication. For example, in a virtual instrumentation application, the DAQ part could be a server for other units, sending messages with measured data to processing clients at request or periodically. Data processing clients could be servers for data presentation devices. In a distributed environment, there may be many interconnected servers and clients, each dedicated to one virtual instrument function (55).

RPC is an abstraction on top of message passing architectures (56). RPC brings a procedural programming paradigm to network programming, adding the abstraction of the function call to distributed systems. In RPC, commu-

nication between programs is accomplished by calling a function on another computer's machine, which creates the illusion that communication with a remote program is not different than communication with a local program.

Distributed object systems extend the idea of RPC with the object-oriented abstraction on top of procedure calls. Distributed object systems supply programs with references to remote objects, which allows the program to control, call methods, and store the remote object in the same way as a local object. The major standard in distributed objects is OMG CORBA, a language-neutral specification for communicating object systems (57). Many standards have been defined on top of CORBA, such as *CORBAMed*, that defines standardized interfaces for health-care objects. Competitors to CORBA include Microsoft's DCOM architecture (58) and the various distributed object systems layered on top of Java (59).

Agent-based integration is potentially an effective integration mechanism. Agent-based systems add the concepts of *autonomy* and *proactivity* to distributed object systems. The agent-oriented approach is well suited for developing complex, distributed systems (60,61). Agents can react asynchronously and autonomously to unexpected situations, increasing robustness and fault-tolerance, which is important in the case of fragile network connections, and for mobile devices (62). As an example of an agent-based distributed integration, we can present a virtual medical device (VMD) agent framework for distributed EEG monitoring with four types of agents: *data agents*, *processing agents*, *presentation agents*, and *monitoring agents* (63). In this framework, data agents abstract data source, creating a uniform view on different types of data, independent of data acquisition device. Processing agents produce derived data, such as power spectrum from raw data provided by the data agents. Presentation agents supply user interface components using a variety of user data views. User interface components are based on HTTP, SMS, and WAP protocols. Monitoring agents collaborate with data and processing agents providing support for data mining operations and search for relevant patterns.

5. TOOLS AND PLATFORMS

In this section, we describe hardware platforms, operating systems, and development environments often used in the development of virtual instruments.

5.1. Hardware Platforms and Operating Systems

Virtual instrumentation and “measurement revolution” is a direct result of another revolution—the PC revolution providing the common hardware platform for virtual instrumentation based on Industry Standard Architecture (ISA) (64). However, other personal computing architectures were also used. For example, LabVIEW 1.0 was developed on a Macintosh computer, which was still supported by LabVIEW. In addition to desktop personal computers, there are more and more pervasive devices such as Internet-enabled cellular phones, PDAs, laptops, and wearable PCs. Although still not often used in virtual

instrumentation, these devices are continually evolving, approaching the capabilities of its desktop counterparts. Therefore, these pervasive devices are increasingly interesting hardware platforms, especially for distributed virtual instrumentation.

Operating systems provide a uniform view on the underlying hardware through the device driver layer, which isolate the details of the sensor interface or sensor device. Commonly used operating systems are Windows operating systems (MS DOS in early days of virtual instrumentation, 95/98/NT/2000/XP/CE), UNIX/Linux, and MacOS.

5.2. Development Environments

Development of virtual instrument is primarily concerned with the development of software, as sensors and hardware are generally available in the open market. We describe two types of virtual instrumentation development environments:

- Conventional programming language environments
- Graphical programming environments

5.2.1. Programming Language Environments. As any other program, software for virtual instrument may be developed with any available general-purpose programming environments. In the late 1970s and early 1980s, BASIC had been the dominant language used with dedicated instrument controllers (1). In the mid- and late 1980s, new programming languages became common, particularly C, as they allowed high-level programming with efficient code. The first version of LabVIEW had been written in C. Nowadays, almost any programming language can be used for development of virtual instruments.

In addition to built-in support of programming languages, developers of virtual instruments often use various third-party software libraries, which in many cases are freely available. For example, the FFTW is the open-source multiplatform library for efficient fast Fourier transform (FFT) analysis (65). Another example is the OpenGL, which allows efficient multiplatform development of effective 3-D graphics presentations.

Java has been a popular implementation environment for various medical and virtual instrumentation solutions because of its architecture and platform independence (66). Various Java toolkits and virtual instrument environments are available (44,55,67). Java-based medical information systems are used to integrate legacy applications for patient records, billing, and pharmacy that are compatible with the industry standard Health Level 7 (HL7) data interchange format. Java also supports all aspects of development of virtual instruments, including work with communication resources, files, databases, Internet communication, multimedia, as well as 2-D and 3-D graphics. Java is also used as a script language for VRML virtual environments (52).

5.2.2. Graphical Programming Tools. Programming environments described in previous sections require from designers and users programming proficiency. New generation of graphical programming tools allows system in-

tegration for ordinary users. Here we describe two such graphical programming tools: Laboratory Virtual Instrument Engineering Workbench (LabVIEW) and BioBench.

5.2.2.1. LabVIEW. National Instruments' LabVIEW made development of virtual instruments more accessible to laboratory users and physicians (68). LabVIEW is the most popular virtual instrumentation tool, and it has been applied to many fields, including virtual bioinstrumentation (29).

LabVIEW was launched with the goal of providing a software tool that empowered engineers to develop customized systems (1). LabVIEW introduced GUIs and visual programming into computerized instrumentation (Fig. 4). LabVIEW is a program development environment, like Java, C, or BASIC. However, although other programming systems use text-based languages to create code, LabVIEW uses a graphical programming language, called G. In LabVIEW, programs are formed as block diagrams. LabVIEW uses the data-flow programming model, in which the execution order is determined by the flow of data between blocks. LabVIEW is also a multitasking and multithreading system.

LabVIEW is a general-purpose programming system with extensive libraries of functions for any programming task. In addition, LabVIEW includes libraries for DAQ, instrument control, data analysis, data presentation, and data storage. It also includes conventional program development tools, such as a debugger; supports many devices and interface standards; has thousands more of built-in analysis, math, and signal processing functions; as well as provides support for SQL and ADO database connectivity, and open connectivity through XML, TCP/IP, wireless, and other standards (28,50).

5.2.2.2. BioBench. BioBench is developed as an extension of LabVIEW for biomedical measurements to simplify

development of biomedical virtual instruments. LabVIEW greatly simplifies programming by introducing the graphical notation, but it still requires a lot of effort to create a virtual instrument. BioBench is primarily designed for physiological DAQ and analysis, for use in research and academic environments (29). It is developed by Premise in collaboration with National Instruments. To execute, the BioBench requires LabVIEW RunTime Engine.

BioBench inherits graphical programming capabilities of LabVIEW while adding customized controls adapted for the measurements of physiological signals such as EEG, ECG, or electromyogram.

6. BIOMEDICAL APPLICATIONS OF VIRTUAL INSTRUMENTATION

Virtual instrumentation has been increasingly accepted in the biomedical field. In relation to the role of a virtual instrument, we may broadly classify biomedical applications of virtual instrumentation into four categories (Fig. 5):

- *Examination*, where a physician does online or offline examination of patient measurements
- *Monitoring*, which can be used as a basis for real-time alerts and interactive alarms
- *Training and education*, where a virtual instrument may simulate or playback measured signals
- *Biofeedback*, where measured signals are presented back to a patient in real time

6.1. Examination

Examination systems are open-loop systems that detect biomedical information from a patient and present it to a physician. During the examination procedure, a physician performs various online or offline analysis of measured

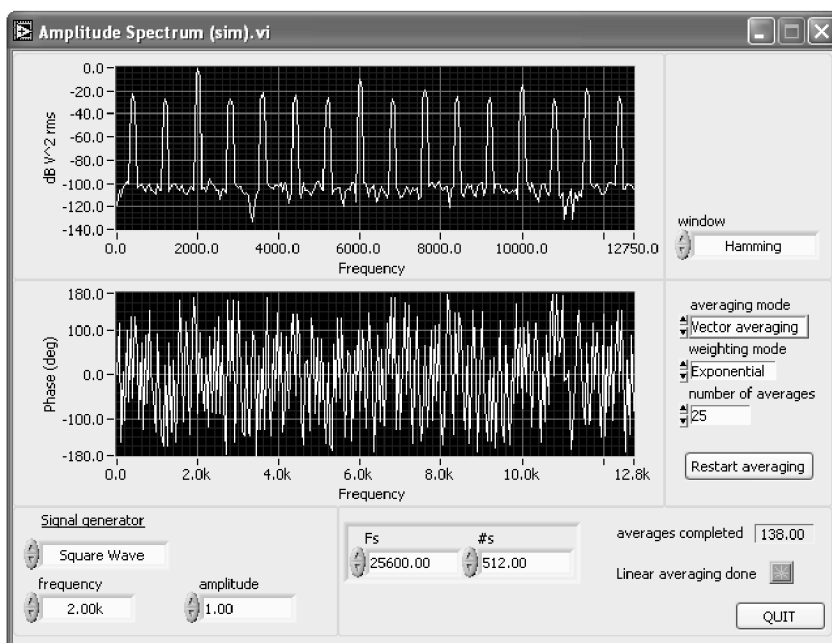


Figure 4. An example of the LabVIEW virtual instrument interface.

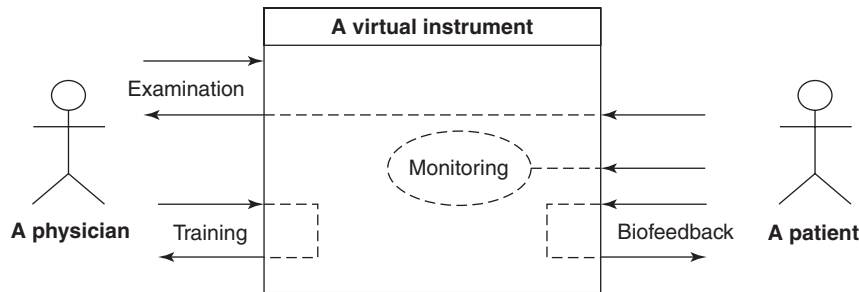


Figure 5. Types of biomedical applications of virtual instrumentation.

data to make a diagnosis (5). Examination can be made locally, in direct contact with a patient, or remotely, where a sensor part is on the patient side connected to a physician through a telecommunication network (46). Nowadays, virtual instrumentation solutions are becoming a part of standard medical examination procedures, with medical systems implemented as virtual instruments.

Many active research projects explore biomedical applications of virtual instrumentation (29). Some examples include canine cardiovascular pressure measurements, cardiopulmonary dynamics measurements, or examination of spontaneous cardiac arrhythmia (69). Advances in cardiology also make possible the design of novel analysis and visualization tools (70,71).

Some other examples include a virtual instrumentation evaluation system for fiberoptic endoscopes (72), PC-based noninvasive measurement of the autonomic nervous system used to detect the onset of diabetic autonomic neuropathy (73), or 3-D posture measurement in dental applications (74).

6.2. Monitoring

Monitoring is a task in which some process continually tracks the measured data, does some analysis, and acts upon detection of some pattern. Monitoring systems are also open-loop systems, as the patient is just an object of monitoring. But in contrast to examination systems, they are more autonomous. Design of monitoring systems is a complex process, as many real-time requirements have to be fulfilled. Integrated with a hospital information system, monitoring can be used as a basis for real-time alerts and interactive alarms (29).

Monitoring and analysis of biomedical signals are also used outside the biomedical field, for example, in affective computing. Physiological parameters that are good indicators of excitement, such as skin conductance and heart rate, are integral data sources for emotional-state-related interactive computer systems (44).

6.3. Training and Education

Virtual instrumentation offers great possibilities for education and improving the skills of physicians. Computer-generated models allow education and training of an operator without actual sensors, which can greatly reduce cost and duration of training (75,76). As the same virtual instrument can work online, play back earlier measured data, or simulate any clinical situation, the training ex-

perience may not differ significantly from the real-world measurements.

Virtual instrumentation may also be integrated with many virtual reality-based applications for education and training. For example, Hofman et al. developed VisualizeR, a virtual environment designed to support the teaching and learning of subjects that require understanding of complex 3-D structures, such as human anatomy (77).

6.4. Biofeedback

Biofeedback systems are closed-loop systems that detect biomedical changes and present them back to the patient in real time to facilitate the change of a user's state. For example, physical rehabilitation biofeedback systems can amplify weak muscle signals, which encourages patients to persist when there is a physical response to therapy that is generally not visible (44). Interfaces in existing biofeedback applications range from interactive 2-D graphical tasks—in which muscle signals are amplified and transformed into control tasks such as lifting a virtual object or typing—to real-world physical tasks such as manipulating radio-controlled toys (8).

Figure 6 shows a multimodal interface for a simple EEG-based biofeedback system. A position of the needle on the display is a function of the calculated EEG parameter, for example the relative power of Alpha activity or ratio of Theta and Beta power. The system incorporates several processing components, including power spectrum computation, statistical functions, and mathematical operations. Before a training session, a physician records the patient's EEG that represents the normal state. During the session, the difference between the patient's current state and the prerecorded state is shown on a display as a change in the gauge position. The subjects are trained to move the needle or to keep it around a desired value. Mapping between EEG values and the needle position is made by user-defined scripts, where it is possible to define various analytic transformations. It is also possible to define which EEG channels will be included, or to scale each channel with different values.

Health-care providers use brain-wave biofeedback as part of the treatment of a growing range of psychophysiological disorders such attention deficit/hyperactivity disorder (ADHD), posttraumatic stress disorder, addictions, anxiety, and depression. In these applications, surface-mounted electrodes detect the brain's electrical activity, and the resulting EEG is presented in real time as abstract images. Using this data in reward/response-based

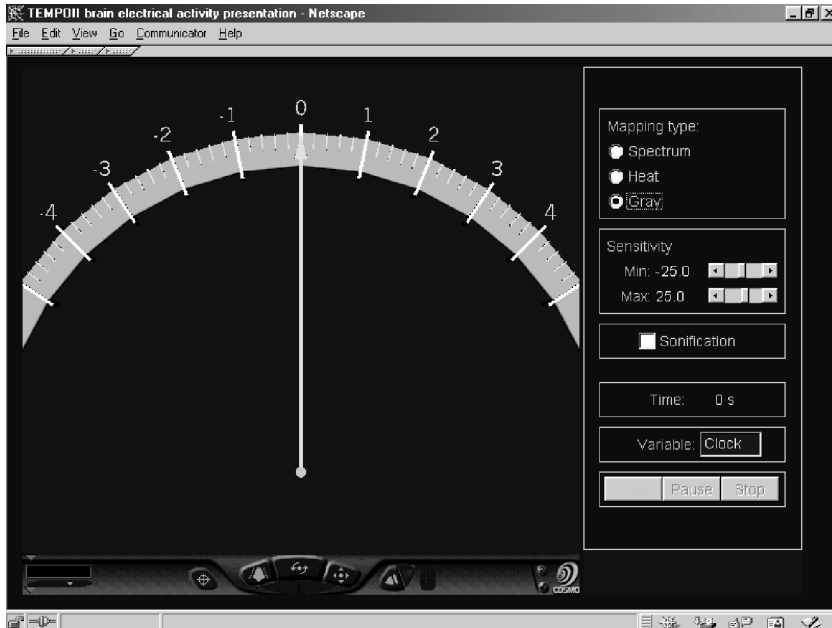


Figure 6. A multimodal interface for simple EEG-based biofeedback system.

control tasks generates increased or reduced activity in different aspects of the EEG spectrum to help ameliorate these psychophysiological disorders (44,78).

7. CONCLUSION

Virtual instrumentation brings many advantages over “conventional” instrumentation. Virtual instruments are realized using industry-standard multipurpose components, and they depend very little on dedicated hardware. Generally, virtual instruments are more flexible and scalable as they can be easily reconfigured in software. Moreover, standard interfaces allow seamless integration of virtual instruments in distributed system. Virtual instrumentation significantly decreases the price of an instrument, as it is based on mass-produced general-purpose computing platforms and dedicated sensors for a given application. We expect an increased number of hardware and software modules designed for the virtual instrumentation market (2). They will provide building blocks for the next generation of instrumentation and measurement. It would not be a surprise if the prefix *virtual* soon disappeared, as virtual instrumentation becomes commonplace.

Virtual instrumentation is rapidly entering the biomedical field. Many general virtual instrumentation concepts may be directly used in biomedical measurements, but biomedical measurements have their specific features that must be taken into account (5). Therefore, although widely used in many biomedical solutions, the virtual instrumentation is not common in critical clinical applications. Having in mind the complexity of biomedical phenomena, bringing virtual instrumentation closer to critical biomedical applications will require more testing and a more extensive list of developed solutions (4). However, according to the current trend, we will not be waiting long for this to happen.

BIBLIOGRAPHY

1. M. Santori, An instrument that isn't really. *IEEE Spectrum* 1990; **27**(8):36–39.
2. H. Goldberg, What is virtual instrumentation? *IEEE Instrum. Measure. Mag.* 2000; **3**(4):10–13.
3. P. J. Denning, Origin of virtual machines and other virtualities. *IEEE Ann. History Comput.* 2001; **23**(3):73.
4. W. Loob, Virtual instrumentation boosts capabilities and reduces costs. *Med. Device Diagnostic Industry Mag.* September 2000: 10.
5. M. Parvis and A. Vallan, Medical measurements and uncertainties. *IEEE Instrum. Measure. Mag.* 2002; **5**(2):12–17.
6. L. A. Geddes and L. E. Baker, *Principles of Applied Biomedical Instrumentation*. 3rd ed. New York: John Wiley & Sons, 1989.
7. F. Nebeker, Golden accomplishments in biomedical engineering. *IEEE Eng. Med. Biol. Mag.* 2002; **21**(3):17–47.
8. J. Charles, Neural interfaces link the mind and the machine. *Computer* 1999; **32**(1):16–18.
9. J. D. Enderle, S. M. Blanchard, and J. D. Bronzino, eds., *Introduction to Biomedical Engineering*. New York: Academic Press, 2000.
10. A. E. Tracht, Adapting laptops to data acquisition. *IEEE Spectrum* 1993; **30**(10):45–47.
11. P. Bhagwat, Bluetooth: Technology for short-range wireless apps. *IEEE Internet Comput.* 2001; **5**(3):96–103.
12. L. Schwiebert, S. K. S. Gupta, and J. Weinmann, Research challenges in wireless networks of biomedical sensors. Proc. 7th Annu. Int. Conf. Mobile Computing and Networking. New York: ACM Press, 2001: 151–165.
13. E. Jovanov, J. Price, D. Raskovic, K. Kavi, T. Martin, R. Adhami, Wireless personal area networks in telemedical environment. Proc. 3rd IEEE EMBS Information Technology Applications in Biomedicine—Workshop of the International Telemedical Information Society ITAB-ITIS 2000, Arlington, VA, 2000: 22–27.

14. E. Jovanov, D. Raskovic, J. Price, A. Moore, J. Chapman, and A. Krishnamurthy, Patient monitoring using personal area networks of wireless intelligent sensors. Proc. 38th Annu. Rocky Mountain Bioengineering Symp., Copper Mountain, CO, April 2001.
15. IEEE Standard for Medical Device Communications — Physical Layer Interface — Cable Connected, IEEE Std 1073.4.1, 2000. Available: <http://standards.ieee.org/catalog/olis/meddev.html>.
16. R. J. Kennelly, The IEEE 1073 standard for medical device communications. Proc. IEEE Systems Readiness Technology Conf. AUTOTESTCON '98, 1998: 335–336.
17. M. Akay, ed., *Time Frequency and Wavelets in Biomedical Signal Processing*. New York: Wiley-IEEE Press, 1997.
18. O. J. Dunn and V. A. Clark, *Basic Statistics: A Primer for Biomedical Sciences*, 3rd ed. New York: John Wiley & Sons, 2000.
19. C. Alippi, A. Ferrero, and V. Piuri, Artificial intelligence for instruments and measurement applications. *IEEE Instrum. Measure. Mag.* 1998; **1**(2):9–17.
20. F. Russo, Fuzzy systems in instrumentation: Fuzzy signal processing. *IEEE Trans. Instrum. Measure.* 1996; **45**(2):683–689.
21. A. Bernieri, G. Betta, A. Pietrosanto, and C. Sansone, A neural network approach to instrument fault detection and isolation. *IEEE Trans. Instrum. Measure.* 1995; **44**(3):747–750.
22. M. Akay, ed., *Nonlinear Biomedical Signal Processing*, vol. 1, Fuzzy Logic, Neural Networks, and New Algorithms. New York: Wiley-IEEE Press, 2000.
23. D. L. Hudson and M. E. Cohen, *Neural Networks and Artificial Intelligence for Biomedical Engineering*. New York: Wiley-IEEE Press, 1999.
24. X. Liu, G. Cheng, and J. X. Wu, AI for public health: Self-screening for eye diseases. *IEEE Intelligent Syst.* 1998; **13**(5):28–35.
25. D. G. Kilman and D. W. Forslund, An international collaborative based on virtual patient records. *Commun. ACM* 1997; **40**(8):111–117.
26. J. Roy and A. Ramanujan, XML: Data's universal language. *IT Professional* 2000; **2**(3):32–36.
27. World Wide Web Consortium (2003). World Wide Web Consortium Web Site (online). Available: <http://www.w3.org/>.
28. National Instruments. (2003). NI LabVIEW Web Site (online). Available: <http://ni.com/labview>.
29. J. B. Olansen and E. Rosow, *Virtual Bio-Instrumentation: Biomedical, Clinical, and Healthcare Applications in LabVIEW*. Englewood Cliffs, NJ: Prentice-Hall, 2001.
30. E. Jovanov, D. Starčević, A. Samardžić, A. Marsh, and Z. Obrenovic, EEG analysis in a telemedical virtual world. *Future Generation Computer Syst.* 1999; **15**:255–263.
31. R. E. Filman, Medicine on the Net: From isolation to universal connectivity. *IEEE Internet Comput.* 2001; **5**(3):40–41.
32. Z. Obrenovic, D. Starcevic, E. Jovanov, and V. Radivojevic, Implementation of virtual medical devices in internet and wireless cellular networks. In: Wojciech Cellary and Arun Iyengar, eds., *Internet Technologies, Applications and Social Impact*, IFIP TC6/WG6.4 Workshop on Internet Technologies, Applications and Social Impact (WITASI 2002), IFIP Conf. Proc. 232. New York: Kluwer, 2002: 229–242.
33. R. A. Robb, *Biomedical Imaging, Visualization, and Analysis*. New York: John Wiley & Sons, 1999.
34. E. Jovanov, D. Starcevic, and V. Radivojevic, Perceptualization of biomedical data. In: M. Akay and A. Marsh, eds., *Information Technologies in Medicine*, vol. 1, Medical Simulation and Education. New York: Wiley-IEEE Press, 2001: 189–204.
35. S. Oviatt and P. Cohen, Perceptual user interfaces: Multimodal interfaces that process what comes naturally. *Commun. ACM* 2000; **43**(3):45–53.
36. G. Mayer-Kress. (2004). Sonification of Multiple Electrode Human Scalp Electro-encephalogram (EEG) (online). Available: <http://www.ccsr.uiuc.edu/People/gmk/Projects/EEG-Sound/>.
37. S. Noah, R. G. Loeb, and R. Watt, Audible display of patient respiratory data, Technical Report, University of Arizona, 1996.
38. D. Sorid and S. K. Moore, The virtual surgeon. *IEEE Spectrum* 2000; **37**(7):26–31.
39. M. Akay and A. Marsh, eds., *Information Technologies in Medicine*, vol. 1, Medical Simulation and Education. New York: Wiley-IEEE Press, 2001.
40. I. Poupjerev, D. S. Tan, M. Billingshurst, H. Kato, H. Regenbrecht, and N. Tetsutani, Developing a generic augmented-reality interface. *Computer* 2002; **35**(3):44–50.
41. Y. Qingping and C. Butler, An object-oriented model of measurement systems. *IEEE Trans. Instrum. Measure.* 1998; **47**(1):104–107.
42. P. Daponte, L. Nigro, and F. Tisato, Object oriented design of measurement systems. *IEEE Instrum. Measure. Mag.* 1992; **41**(6):874–880.
43. G. R. McMillan, R. G. Eggleston, and T. R. Anderson, Non-conventional controls. In: G. Salvendy, ed., *Handbook of Human Factors and Ergonomics*, 2nd ed., New York: John Wiley & Sons, 1997: 729–771.
44. J. Allanson, Electrophysiologically interactive computer systems, *Computer* 2002; **35**(3):60–65.
45. T. L. Huston and J. L. Huston, Is telemedicine a practical reality? *Commun. ACM* 2000; **43**(6):91–95.
46. D. Starčević, E. Jovanov, V. Radivojević, Z. Obrenovic, and A. Samardžic, Virtual medical devices for telemedical applications. In: P. Spasić, I. Milosavljević, and M. Jancic-Zguricas, eds., *Telemedicine*. Belgrade, Yugoslavia: Academy of Medical Sciences of Serbian Medical Association, 2000, pp. 218–244.
47. G. Fortino and L. Nigro, Development of virtual data acquisition systems based on multimedia internet networking. *Computer Standards Interfaces* 1999; **21**(5):429–440.
48. C. Potter, R. Brady, P. Moran, C. Gregory, B. Carragher, N. Kisseberth, J. Lyding, and J. Lindquist, EVAC: a virtual environment for control of remote imaging instrumentation. *IEEE Comput. Graphics Applicat.* 1996; **16**(4):62–66.
49. K. Goldberg, Introduction: The unique phenomenon of a distance. In: K. Goldberg, ed., *The Robot in the Garden: Teletronics and Telepistemology in the Age of the Internet*. Cambridge, MA: MIT Press, 2000.
50. J. Travis, *Internet Applications in LabVIEW*. Englewood Cliffs, NJ: Prentice-Hall, 2000.
51. Z. Obrenovic, D. Starcevic, E. Jovanov, and V. Radivojevic, An implementation of real-time monitoring and analysis in telemedicine. Proc. 3rd IEEE EMBS Information Technology Applications in Biomedicine—Workshop of the International Telemedical Information Society ITAB-ITIS 2000, Arlington, VA, 2000: 74–78.
52. The VRML Consortium Incorporated, Information technology—Computer graphics and image processing—The Virtual Reality Modeling Language (VRML)—Part 1: Functional specification and UTF-8 encoding. International Standard

- ISO/IEC 14772-1:1997, 1997. (2003) (online). Available: <http://www.vrml.org/Specifications/VRML97/index.html>.
53. B. Davie, V. Florance, A. Friede, J. Sheehan, and J. E. Sisk, Bringing health-care applications to the internet. *IEEE Internet Comput.* 2001; **5**(3):42–48.
 54. N. Leavitt, Will WAP deliver the wireless internet? *Computer* 2000; **33**(5):16–20.
 55. D. Grimaldi, L. Nigro, and F. Pupo, Java-based distributed measurement systems. *IEEE Trans. Instrum. Measure.* 1998; **47**(1):100–103.
 56. Network Working Group Sun Microsystems. RPC: Remote Procedure Call Protocol Specification. RFC 1057 (June 1988) (online). Available: <http://info.internet.isi.edu:80/in-notes/rfc/files/rfc1057.txt>.
 57. Object Management Group (OMG), The Common Object Request Broker: Architecture and Specification (CORBA), revision 2.0. Object Management Group (OMG), 2.0 ed.
 58. R. Sessions, *COM and DCOM: Microsoft's Vision for Distributed Objects*. New York: John Wiley & Sons, 1997.
 59. Sun Microsystems, RMI: Remote Method Invocation Web Site. (2002) (online). Available: <http://java.sun.com/products/jdk/rmi/index.html>.
 60. N. R. Jennings, An agent-based approach for building complex software systems. *Commun. ACM* 2001; **44**(6):35–41.
 61. Y. I. Wijata, D. Niehaus, and V. S. Frost, A scalable agent-based network measurement infrastructure. *IEEE Commun. Mag.* 2000; **38**(9):174–183.
 62. D. B. Lange and M. Ochima, Seven good reasons for mobile agents. *Commun. ACM* 1999; **42**(3):88–89.
 63. Z. Obrenovic, D. Starcevic, E. Jovanov, and V. Radivojevic, An agent based framework for virtual medical devices. Proc. 1st Int. Joint Conf. Autonomous Agents and Multiagent Systems AAMAS 2002. New York: ACM Press, 2002: 659–660.
 64. National Instruments (1999). The Measurement Revolution, White Paper (online). Available: <http://www.ni.com/revolution/learn.htm>.
 65. M. Frigo and S. G. Johnson, FFTW: An adaptive software architecture for the FFT. 1998 ICASSP Conf. Proc. vol. 3, pp. 1381–1384 (online). Available: <http://www.fftw.org/>.
 66. T. Jepsen, Don't know beans? Try Java. *IT Professional* 2000; **2**(2):29–35.
 67. F. Zubillaga-Elorza and C. R. Allen, Virtual instrument toolkit: Rapid prototyping on the Web. *IEEE Internet Comput.* 1999; **3**(5):41–48.
 68. N. Ertugrul, *LabVIEW for Electric Circuits, Machines, Drives, and Laboratories*. Englewood Cliffs, NJ: Prentice-Hall PTR, 2002.
 69. D. L. Rollins, C. R. Killingsworth, G. P. Walcott, R. K. Justice, R. E. Ideker, and W. M. Smith, A telemetry system for the study of spontaneous cardiac arrhythmias. *IEEE Trans. Biomed. Eng.* 2000; **47**(7):887–892.
 70. J. F. Thayer, Progress in heart-rate variability analysis. *IEEE Eng. Med. Biol. Mag.* 2002; **21**(4):22–23.
 71. E. Tataru and A. Cinar, Interpreting ECG data by integrating statistical and artificial intelligence tools. *IEEE Eng. Med. Biol. Mag.* 2002; **21**(1):36–41.
 72. E. Rosow, J. Adam, F. Beatrice, K. Youngdahl, and R. Orlando, A virtual instrumentation evaluation system for fiberoptic endoscopes. Presented at the Society of American Gastrointestinal Endoscopic Surgeons (SAGES), March 21, 1997, San Diego, CA.
 73. S. Pruna and A. Dumitrescu, PC-based noninvasive measurement of the autonomic nervous system. Detecting the onset of diabetic autonomic neuropathy. *IEEE Eng. Med. Biol. Mag.* 1998; **17**(2):66–72.
 74. M. Baszynski, C. R. Allen, Z. Moron, and P. A. Heasman, 3D posture measurement in dental applications. IEEE Int. Symp. Virtual Environments and Measurement Systems VIMS '02, May 2002, Paper VM-2017.
 75. J. L. Schmalzel, Instrumentation education. *IEEE Instrum. Measure. Mag.* 1998; **1**(1):34–35.
 76. J. C. Waller and N. Foster, Training via the web: A virtual instrument. *Comput. Educ.* 2000; **35**(2):161–167.
 77. H. Hoffman, M. Murray, R. Curlee, and A. Fritchle, Anatomic VisualizeR: Teaching and learning anatomy with virtual reality. In: M. Akay and A. Marsh, eds., *Information Technologies in Medicine*. vol. 1, Medical Simulation and Education. New York: Wiley-IEEE Press, 2001, pp. 205–218.
 78. M. S. Schwartz and F. Andrasik, eds., *Biofeedback: A practitioner's guide*. New York: The Guilford Press, 2003.

KEYWORDS

virtual instrumentation, computerized instrumentation, biomedical measurements, biomedical applications

ABSTRACT

Virtual instrumentation merges sensing technologies with hardware and software technologies to create flexible and sophisticated instruments for various control and monitoring applications. Biomedical applications require sophisticated and flexible instrumentation, accomplished by using general-purpose computing platforms with various application-specific input/output devices. Virtual instrumentation brings many advantages over “conventional” instrumentation. Employing general-purpose computing platforms significantly decreases the price of instruments. Standard system interfaces allow seamless integration of virtual instruments in a distributed system, whereas software reconfiguration facilitates flexibility and scalability. Most general virtual instrumentation concepts are directly applicable in biomedical applications; however, specific features of the biomedical instrumentation must be taken into account. In this article, we present a brief history and current state of virtual instrumentation for biomedical applications. We explain the general architecture of a virtual instrument, describe the most frequently used tools and platforms, and present typical examples of biomedical applications of virtual instrumentation.

Author Query Form



Title: Encyclopedia of Biomedical Engineering
Article/Number: Virtual Instrumentation/1265

Dear Author,

During the preparation of your manuscript for typesetting some questions have arisen. These are listed below. Please check your typeset proof carefully and mark any corrections in the margin of the proof or compile them as a separate list. This form should then be returned with your marked proof/list of corrections to John Wiley.

This form should then be returned with your marked proof/list of corrections to N.S. Pandian, Deputy Manager, Macmillan India Ltd., Book Division–IPD, Midford Crescent, New No. 159/1 (Old No. 53/1), Richmond Road, Bangalore-560025, India, Tel: +91-80-51237312; +91-80-51237313, Fax: +91-80-51237310, E-mail: ns.pandian@macmillan-india.co.in

Queries and/or remarks

AU:1	The spelling is not matching with the reference. Please verify & confirm.
------	---