Tesis Doctoral
2013

Laboratory as a Service (LaaS): a Paradigm for Developing and Implementing Modular Remote Laboratories

---

Laboratorio como Servicio (LaaS): un Paradigma para Desarrollar e Implementar Laboratorios Remotos Modulares

Mohamed Saied Tawfik Abuelela

MSc, Ingeniería Electrónica
Universidad Nacional de Educación a Distancia, Madrid, España (2011)
BSc, Ingeniería Eléctrica
Universidad de Ain Shams, El Cairo, Egipto (2008)

Departamento de Ingeniería Eléctrica, Electrónica, y de Control (DIEEC)

Escuela Técnica Superior de Ingenieros Industriales (ETSII)

Universidad Nacional de Educación a Distancia (UNED)

Director: Manuel Alonso Castro Gil
Catedrático de Universidad
Presidente de la Sociedad de Educación del IEEE

Co-director: Elio Sancristobal Ruiz
Ayudante Doctor

Tesis Doctoral
2013

Laboratory as a Service (LaaS): a Paradigm for Developing and Implementing Modular Remote Laboratories

---

Laboratorio como Servicio (LaaS): un Paradigma para Desarrollar e Implementar Laboratorios Remotos Modulares

Mohamed Saied Tawfik Abuelela

MSc, Ingeniería Electrónica
Universidad Nacional de Educación a Distancia, Madrid, España (2011)
BSc, Ingeniería Eléctrica
Universidad de Ain Shams, El Cairo, Egipto (2008)

Departamento de Ingeniería Eléctrica, Electrónica, y de Control (DIEEC)

Escuela Técnica Superior de Ingenieros Industriales (ETSII)

Universidad Nacional de Educación a Distancia (UNED)

Director: Manuel Alonso Castro Gil
Catedrático de Universidad
Presidente de la Sociedad de Educación del IEEE

Co-director: Elio Sancristobal Ruiz
Ayudante Doctor

Departamento de Ingeniería Eléctrica, Electrónica, y de Control (DIEEC)

Escuela Técnica Superior de Ingenieros Industriales (ETSII)

Universidad Nacional de Educación a Distancia (UNED)

Laboratory as a Service (LaaS): a Paradigm for Developing and Implementing Modular Remote Laboratories

Laboratorio como Servicio (LaaS): un Paradigma para Desarrollar e Implementar Laboratorios Remotos Modulares

Mohamed Saied Tawfik Abuelela

MSc, Ingeniería Electrónica
Universidad Nacional de Educación a Distancia, Madrid, España (2011)
BSc, Ingeniería Eléctrica
Universidad de Ain Shams, El Cairo, Egipto (2008)

Director: Manuel Alonso Castro Gil
Catedrático de Universidad
Presidente de la Sociedad de Educación del IEEE

Co-director: Elio Sancristobal Ruiz
Ayudante Doctor

Acknowledgments

Foremost, my heartfelt thanks go to my parents Shadia and Saied, to my sister Heba, and to my brothers Mahmoud and Ahmed for their unfailing love, patience, prayers, and selfless support. It is to my family that I dedicate this thesis.

My sincere gratitude and appreciation go to my personal and professional value adding supervisor, Professor Manuel Castro, who has nurtured my passion for science and knowledge throughout my PhD years. Words are inadequate to express my gratitude to this great person for his tremendous motivation, support, and continuous advice and encouragement.

I would also like to acknowledge the appreciation of my co-supervisor, Professor Elio Sancristobal, who has been very helpful and supportive of me. Similarly, to my colleagues, the entire members of DIEEC-UNED—too numerous to mention here, who made my experience here in Madrid and at UNED fulfilling.

My deepest and affectionate thanks to Professor David Lowe for his consistent guidance and inspiring discussions during my research visit at University of Technology, Sydney (UTS). As well, to all his research group for their technical support. Similarly, to Professor Christophe Salzmann and Professor Denis Gillet, for the insightful research advices and guidance I had the privilege to receive during my research visit at École Polytechnique Fédérale de Lausanne (EPFL) and during the writing of this thesis. Thanks David, Christophe, and Denis!

I am genuinely indebted to the Spanish Ministry of Economy and Competitiveness for funding my PhD through its FPI pre-doctoral scholarships grant (BES-2009-028307), which was linked with the sLab project (*TIN2008-06083-C03/TSI*). I would also like to acknowledge the financial support of the following research projects: SOLITE (*CYTED-P507AC0526*), mPSS (*142788-2008-BG-LEONARDO-LMP*), e-Madrid (*S2009/TIC-1650*), BEOL (NSF-1132813), PAC (*517742-LLP-1-2011-1-BG-ERASMUS-ECUE*), RIPLECS (*517836-LLP-1-2011-1-ES-ERASMUS-ESMO*), EMTM (*2011-1-PL1-LEO05-19883-ESMO*), MUREE (*530332-TEMPUS-1-2012-1-JO-TEMPUS-JPCR*), and GO-Lab (*530332-TEMPUS-1-2012-1-JO-TEMPUS-JPCR*).

I would like to extend my utmost gratitude to the founders and developers of the Virtual Instrument Systems in Reality (VISIR) project at Blekinge Institute of Technology (BTH), Professor Ingvar Gustavsson, Professor Lars Ha°kansson, Johan Zackrisson, and Kristian Nilsson. As well, I am grateful to other research groups within the Global Online Laboratory Consortium (GOLC): Professor Javier Garcia-Zubia and his research group from University of Deusto (UD), Professor Gustavo Alves and his research group from Instituto Superior de Engenharia do Porto (ISEP), Professor Judson Harward and his research group from Massachusetts Institute of Technology (MIT), and Professor Michael Auer and his research group from Carinthia University for Applied science (CUAS). Last but not least, I would like to thank Professor Hamadou Saliah-Hassane and the IEEE P1876™ (Standard for Networked Smart Learning Objects for Online Laboratories) working group.

Mohamed Tawfik
Madrid, Spain
October 2013

Laboratory as a Service (LaaS): a Paradigm for Developing and Implementing Modular Remote Laboratories

by

Mohamed Saied Tawfik Abuelela

Doctoral thesis submitted to the Electrical & Computer Engineering Department of the Spanish University for Distance Education (UNED) on October 2013, in partial fulfillment of the requirements of the degree of Doctor of Philosophy (PhD)

Abstract

This thesis introduces a novel paradigm, Laboratory as a Service (LaaS), for developing modular remote laboratories—based on independent component modules—and implementing them as a set of loosely-coupled services to be consumed with a high level of abstraction and virtualization. The main goals are to: (1) define an organized manner for sharing remote laboratories globally among institutions; (2) allow wrapping remote laboratories in any heterogeneous application container, as well as, their coupling and mashing up with heterogeneous services across the Web; (3) facilitate maintenance, reusability, and leveraging legacy equipment; and (4) allow interchangeability of components between provider and consumer—seamlessly and programmatically—insofar as consumer could contribute with one or more components instead of full-reliance on the provider's equipment and facilities. Beyond the academic context, LaaS will facilitate the incorporation of remote laboratories in the ecosystem of the ubiquitous smart things surrounding us, which increases everyday with the approaching Web of Things (WoT) and artificial intelligence era. This, in turn, will create a breeding ground for online control, experimentation, and discovery—in either formal or informal context and with neither temporal nor geographical constraints. Last but not least, LaaS aims to set principles for a global standardized design pattern to be adopted by remote laboratories developers, providers and consumers.

Key Words: automation, cloud computing, computer-aided engineering, eLearning, grid computing, interoperability, mashup, remote laboratories, Web of Things (WoT).

Thesis Supervisor:	Manuel Alonso Castro Gil Full Professor IEEE Education Society President
Thesis Co-Supervisor:	Elio Sancristobal Ruiz Assistant Professor

Laboratorio como Servicio (LaaS): un Paradigma para Desarrollar e Implementar Laboratorios Remotos Modulares

Mohamed Saied Tawfik Abuelela

Tesis presentada en el Departamento de Ingeniería Eléctrica, Electrónica, y de Control (DIEEC) de la Universidad Nacional de Educación a Distancia (UNED) en Octubre 2013, como parte del requerimiento de la obtención del grado de Doctor (PhD)

Resumen

En esta tesis se introduce un nuevo paradigma, Laboratorio como Servicio (en inglés, “Laboratory as a Service”, y cuyas siglas en inglés son, LaaS), para desarrollar laboratorios remotos modulares—basados en componentes modulares independientes—e implementarlos como un conjunto de servicios de acoplamiento débil con el fin de utilizarse con un nivel alto de abstracción y virtualización. Los objetivos principales son: (1) definir una forma organizada de compartir los laboratorios remotos globalmente; (2) permitir la integración de los laboratorios remotos en cualquier aplicación heterogénea, así como, su acoplamiento con cualquier servicio heterogéneo existente en la Web; (3) facilitar el mantenimiento y la reutilización de los equipos antiguos; y (4) permitir el intercambio de los componentes entre el proveedor y el consumidor de tal manera que el consumidor podría contribuir con un componente o más en vez de depender completamente de los componentes y los equipos del proveedor. Más allá del contexto académico, LaaS facilitará la incorporación de los laboratorios remotos en el ecosistema de los aparatos inteligentes y ubicuos que nos rodean y que se incrementa cada día con la llegada de la era de la inteligencia artificial y de la Web de las Cosas. Esto dará lugar a la experimentación y el descubrimiento en línea—tanto en contexto formal como informal y sin restricciones geográficas ni temporales. Por último, LaaS pretende establecer bases para un modelo global estandarizado para los desarrolladores de los laboratorios remotos, proveedores y consumidores.

Palabras Claves: aplicación Web híbrida, aprendizaje electrónico, automatización, computación en la nube, computación grid, ingeniería asistida por computadora, interoperabilidad, laboratorios remotos, Web de las cosas.

Director:	Manuel Alonso Castro Gil Catedrático de Universidad Presidente de la Sociedad de Educación del IEEE
Co-Director:	Elio Sancristobal Ruiz Ayudante Doctor

To my parents Shadia & Saied, to my
sister Heba, & to my brothers
Mahmoud & Ahmed

"Education is what remains after one has forgotten everything he learned in school"

— Albert Einstein

Table of Contents

Abbreviations **7**

List of Figures **17**

List of Tables **21**

Chapter 1

1. Introduction **23**

1.1. Hands-on Laboratories 27

1.2. Online Laboratories 28

1.3. Motivation 30

1.4. Description 32

1.5. Objectives 32

1.6. Research Methodology 33

1.7. Accomplishments and Contributions 34

1.7.1. Participation in Research Projects 34

1.7.2. Research Visits 36

1.7.3. Assisted Scientific Events and Conferences 37

1.7.4. Published Journal Articles 38

1.7.5. Published Book Chapters 39

1.7.6. Published Conference Papers 40

1.7.7. Other Publications 46

1.7.8. Other Activities and Merits 46

1.8. Thesis Outline 48

Chapter 2

2. Remote Laboratories Applications in Electrical Engineering Education **51**

2.1. Electronics 51

2.1.1. NetLab 53

2.2. Digital Electronics 55

2.3. Microelectronics 56

2.4. Power Systems and Electric Machines 57
2.5. Renewable Energy 61
2.6. Signal Processing 62
2.7. Telecommunications 65
2.8. Instrumentation & Measurements 67
2.9. Control 69
2.10. Industrial Control Systems (ICSs) 71
2.11. Embedded Systems 75
2.12. Systems, Mechatronics, & Robotics 76
2.13. Biomedical Engineering 79
2.14. Multidisciplinary Commercial Solutions 80
2.14.1. NI Educational Laboratory Virtual Instrumentation Suite (ELVIS) II 80
2.14.2. NI myDAQ 83
2.14.3. NI Universal Software Radio Peripheral (USRP) 84
2.15. Conclusion 85

Chapter 3

3. Development and Implementation of Remote Electronics Experiments 89
3.1. Hardware Description 90
3.1.1. PXI Platform 90
3.1.2. Relay Switching Matrix 90
3.2. Software Description and Operation Cycle 96
3.2.1. User Interface (UI) 96
3.2.2. Experiment Client 97
3.2.3. Measurement Server 100
3.2.4. Equipment Server 101
3.3. Deployment and Results 103
3.4. Implementation in a MOOC at UNED 106
3.5. Industrial Electronics Experiments 108
3.5.1. Second-Order Low-Pass RLC Filter 111
3.5.2. Second-Order High-Pass RLC Filter 111
3.5.3. AC/DC Converter (Full-Wave Rectifier) 112

3.5.4. Non-Isolated Linear Regulated DC/DC Converter 113
3.5.5. Non-Isolated Switching Regulated DC/DC Converter 114
3.5.6. Remote Retrieved Results 116
3.6. Implementation in a Master Degree Program 119
3.7. Summary 121

Chapter 4

4. Integration into Educational Systems 123
4.1. Integration into Metadata Repositories 124
4.2. Integration into Learning Management Systems (LMSs) 125
4.3. Integration into Personal Learning Environments (PLEs) 127
4.4. Integration into Remote Laboratory Management Systems (RLMSs) 129
4.5. System-System Communication 136
4.6. Discussion 142

Chapter 5

5. Laboratory as a Service (LaaS) 145
5.1. Background 145
5.2. Features 147
5.3. Objectives 148
5.4. Related Works 149
5.5. Context 150
5.5.1. ELearning 3.0 151
5.5.2. Service-Oriented LMS 154
5.5.3. Mobile Access 157
5.6. LaaS Description 159
5.6.1. Implementation 164
5.7. Modular Architecture 173
5.7.1. Data Acquisition (DAQ) Card 175
5.7.2. Switching Board 175
5.7.3. Controller 178

5.7.4. Automatic Test Equipment (ATE) ... 179
5.7.5. Communication Buses ... 181
5.7.6. Laboratory Server Software ... 190
5.7.7. User Interface (UI) ... 198
5.8. Introductory Examples ... 206
5.8.1. Example 1 ... 206
5.8.2. Example 2 ... 208
5.9. Summary ... 208

Chapter 6

6. Modular Remote Laboratory Prototype ... 213
6.1. Introduction ... 213
6.2. Development ... 216
6.2.1. Hardware ... 216
6.2.2. Laboratory Server Software ... 217
6.2.3. Video Streaming Server ... 234
6.3. Delivery as a Service ... 236
6.4. Consumption ... 238
6.5. Summary ... 243

Chapter 7

7. Conclusions ... 245
7.1. Suggestions for Future Works ... 247
7.1.1. Standardization ... 248
7.2. Summary of Outputs ... 249
7.2.1. Research ... 249
7.2.2. Development and Implementation ... 250
7.2.3. Dissemination and Publications ... 251
7.2.4. Other Merits ... 251

Appendix A

A. CD Content 253

Appendix B

B. Resumen (Español) 255
B.1. Introducción 255
B.2. Motivación 257
B.3. Solución Propuesta 258
B.4. Objetivos 259
B.5. Metodología 260
B.6. Estructura de la Tesis 261
B.7. Conclusión y Sugerencias para Trabajos Futuros 264
B.7.1. Estandarización 265
B.8. Resumen de los Resultados 266
B.8.1. Investigación 266
B.8.2. Desarrollo e implementación 268
B.8.3. Diseminación y Publicaciones 269
B.8.4. Otros Méritos 269

Appendix C

C.VISIR Configurations 271
C.1. Hardware 271
C.2. Software 272
C.3. Datasheets 279

Appendix D

D. ReLaSIS 285

Appendix E

E. Relevant Merits 287

Appendix F

F. Additional References...299

References...301

Abbreviations

ABET	Accreditation Board for Engineering and Technology
ANC	Active Noise Control
ADO	ActiveX Data Object
AdvancedTCA	Advanced Telecom Computing Architecture
AC	Alternating Current
EC2	Amazon Elastic Compute Cloud
AM	Amplitude Modulation
ASK	Amplitude-Shift Keying
AI	Analog Input
AO	Analog Output
A/D	Analog to Digital
XaaS	Anything as a Service
API	Application Programming Interface
ASIC	Application-Specific Integrated Circuit
AOU	Arab Open University
AWG	Arbitrary Waveform Generator
SDM	are Space-Division Multiplexing
ALU	Arithmetic Logic Unit
AJAX	Asynchronous JavaScript and XML
ATE	Automatic test equipment
BIOS	Basic I/O System
BNC	Bayonet Neill–Concelman
BJT	Bipolar Junction Transistor
BTH	Blekinge Tekniska Högskola
B-Learning	Blended Learning
BRPMN	Business Process Model and Notation
B2B	Business-to-Business
CSS	Cascading Style Sheets
CPU	Central Processing Unit
CDM	Code Division Multiplexing
COTS	Commercial Off-The-Shelf
CLS	Common Language Specification
CORBA	Common Object Request Broker Architecture

CMOS	Complementary Metal-Oxide-Semiconductor
CPLD	Complex Programmable Logic Device
CBT	Computer-Based Training
cMOOC	connectivist-based MOOC
xMOOCs	content-based MOOC
CAN	Controller Area Network
CRUD	Create, Read, Update, and Delete
DAQ	Data Acquisition
DDS	Data Distribution Service
DOF	Degree of Freedom
DEMUX	demultiplexer
DUT	Device under Test
DPWS	Devices Profile for Web Services
DIO	Digital Input/Output
DSP	Digital Signal Processing or Digital Signal Processor
D/A	Digital to Analog
DC	Direct Current
DSSS	Direct-Sequence Spread Spectrum
DCOM	Distributed Component Object Model
DCE	Distributed Computing Environment
DCS	Distributed Control System
DNP3	Distributed Network Protocol 3
DOM	Document Object Model
DPDT	Double Pole Double Throw
DPDT	Double Pole Double Throw
DSB	Double-Sideband
DIP	Dual in-Line Package
DHTML	Dynamic HTML
DLL	Dynamic Link Library
DSA	Dynamic Signal Analyzer
EJS	Easy Java Simulation
EPFL	École Polytechnique Fédérale de Lausanne
ELF	E-Learning Framework
EEG	Electroencephalography
EMG	Electromyography
EAI	Enterprise application integration
ESS	Experiment Storage Service
XHTML	Extensible HyperText Markup Language

XML	Extensible Markup Language
CUAS	Fachhochschule Kärnten
FET	Field-Effect Transistor
FPGA	Field-Programmable Gate Array
FIR	Finite Impulse Response
FM	Frequency Modulation
FDM	Frequency-Division Multiplexing
FSK	Frequency-Shift Keying
FOAF	Friend of a Friend
FSF	Full State Feedback
FBD	Function Block Diagram
GPIB	General Purpose Interface Bus
GAL	Generic Array Logic
GOLC	Global Online Laboratory Consortium
Go-Lab	Global Online Science Labs for Inquiry Learning at School
GPS	Global Positioning System
GPS	Global Positioning System
GPS	Global Positioning System
GSM	Global System for Mobile Communications
DPL	GNU General Public License
GCE	Google Compute Engine
GUI	Graphical User Interface
HDL	Hardware Description Language
HIL	Hardware-in-the-Loop
HDVC	High Voltage Direct Current
HTTPS	HTTP Secure
HMI	Human–Machine Interface
HTML	Hypertext Markup Language
HTTPS	Hypertext Transfer Protocol
ISA	iLab Shared Architecture
IMS AF	IMS Abstract Framework
QTI	IMS Question and Test Interoperability
IMS TI	IMS Tool Interoperability
ICS	Industrial Control System
ISM	Industrial Scientific and Medical
INS	Inertial Navigation System
IIR	Infinite Impulse Response
ICT	Information and Communication Technologies

IaaS	Infrastructure as a Service
ISEP	Instituto Superior de Engenharia do Porto
IL	Instruction List
ISP	In-System Programming
IC	Integrated Circuits
IDE	Integrated Development Environment
ISE	Integrated Software Environment
IVI	Interchangeable Virtual Instruments
IDL	Interfaces Definition Language
I2C	Inter-Integrated Circuit
IAOE	International Association of Online Engineering
IBM	International Business Machines Corporation
IEC	International Electrotechnical Commission
SI	International System of Units
IIOP	Internet InterORB Protocol
IoT	Internet of things
IBT	Internet-Based Training
JBI	Java Business Integration
JDBC	Java Database Connectivity
RMI	JAVA Remote Method Invocation
JRMP	Java Remote Method Protocol
JRE	Java Runtime Environment
JVM	Java Virtual Machines
JIL	Java-Internet-LabVIEW
JIM	Java-Internet-MATLAB
JSON	JavaScript Object Notation
JTAG	Joint Test Action Group
LSS	Lab Side Scheduling Service
Laas	Laboratory as a Service
LabVIEW	Laboratory Virtual Instrumentation Engineering Workbench
LD	Ladder diagram
LDL	Lab Description Language
LXI	LAN eXtensions for Instrumentation
LSI	Large-Scale Integration
LMS	Learning Management System
LOM	Learning Object Metadata
LMS	Least Mean Square
LEDs	Light-Emitting Diodes

LDAP	Light-weight Directory Access Protocol
LLO	LiLa Learning Object
LQR	Linear-Quadratic Regulator
LCD	Liquid-Crystal Display
LAN	Local Area Network
LON	Local Operating Network
MRI	Magnetic Resonance Imaging
MEF	Managed Extensibility Framework
MIT	Massachusetts Institute of Technology
MOOC	Massive Open Online Course
MTU	Master Terminal Unit
MCR	MATLAB Compiler Runtime
MAC	Media Access Control
MRL	Media Resource Locator
MTOM	Message Transmission Optimization Mechanism
MOSFET	Metal–Oxide–Semiconductor Field-Effect Transistor
MCU or µC	Microcontroller
ADO	Microsoft ActiveX Data Objects
MSDOS	Microsoft Disk Operating System
COM	Microsoft's Component Object Model
M-Learning	Mobile Learning
MIMO	Multi-Input and Multi-Output
DMM	multimeter
MUX	Multiplexer
MXI	Multisystem eXtension Interface
NI	National Instruments
NSF	National Science Foundation
NTC	Negative Temperature Coefficient
NETBSD	NET Berkeley Software Distribution
NI ELVIS II	NI Educational Laboratory Virtual Instrumentation Suite II
NI RIO	NI reconfigurable I/O
OLEDB	Object Linking and Embedding Database
OMG	Object Management Group
ORB	Object Request Broker
OPC UA	OLE for Process Control Unified Architecture
ODBC	Open Database Connectivity
OER	Open Educational Resources
OKI	Open Knowledge Initiative

OSGi	Open Service Gateway initiative
OSID	Open Service Interface Definition
OSGi	Open Services Gateway initiative
OUA	Open Universities Australia
OU	Open University
OCW	OpenCourseWare
OS	Operating System
OSE	Operating System Embedded
Oracle IaaS	Oracle Infrastructure as a Service
ODE	Ordinary Differential Equation
PCIe	PCI Express
PXI	PCI eXtensions for Instrumentation
PCI	Peripheral Component Interconnect
PIC	Peripheral Interface Controller
PM	Permanent Magnet
PC	Personal Computer
PDA	Personal Digital Assistants
PLE	Personal Learning Environment
PLE	Personal Learning Environment
PM	Phase Modulation
PSK	Phase-Shift Keying
PCG	Phonocardiogram
PV	Photovoltaic
PaaS	Platform as a Service
PTP	Precision Time Protocol
PAL	Programmable Array Logic
PFI	Programmable Function Interface
PLC	Programmable Logic Controllers
PLD	Programmable Logic Device
PSoC	Programmable SoC
PID	Proportional Integral Derivative
PN	Pseudorandom Number
PWM	Pulse Width Modulation
PAM	Pulse-Amplitude Modulation
PCM	Pulse-Code Modulation
PXIe	PXI Express
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service

IDDQ	quiescent supply current
RF	Radio Frequency
RFI	Radio Frequency Interference
RFID	Radio-frequency identification
RAM	Random-Access Memory
ROM	Read-Only Memory
RTOS	Real-Time OS
RTPS	Real-Time Publish Subscribe
RDMS	Relational Database Management System
RDMS	Relational Database Management Systems
RDP	Remote Desktop Protocol
REV	Remote Engineering & Virtual Instrumentation
RLMS	Remote Laboratory Management Systems
ReLaSIS	Remote Laboratory System Interface Specification
RPC	Remote Procedure Calls
RTU	Remote Terminal Unit
REST	Representational State Transfer
RTD	Resistance Temperature Detector
RDF	Resource Description Framework
RIA	Rich Internet Application
STEM	Science, Technology, Engineering, and Mathematics
SSH	Secure Shell
SSL	Secure Sockets Layer
SFC	Sequential Function Chart
SCI	Serial Communication Interface
SCA	Service Component Architecture
SOA	Service Oriented Architecture
SOA	Service Oriented Architecture
SSD	Seven-Segment Display
SCO	Sharable Content Object
SCORM	Shareable Courseware Object Reference Model
Rsh	shunt resistor
SNR	Signal-to-Noise Ratio
SMTP	Simple Mail Transfer Protocol
SOAP	Simple Object Access Protocol
SISO	Single-Input and Single-Output
SPDT	Single-Pole Double-Throw
SPST	Single-Pole Single-Throw

SSB	Single-Sideband
SaaS	Software as a Service
SDK	Software Development Kit
SIG VISIR	Special Interest Group of VISIR
SCPI	Standard Commands for Programmable Instruments
ST	Structured Text
SAF	Stuck-at Faults
SOP	Sum-of-Products
SCADA	Supervisory Control and Data Acquisition
SCC	Synchronous Serial Communication
SoC	System on Ship
TEL	Technology-Enhanced Learning
T&M	Test and Measurement
EHEA	the European Higher Education Area
TDMA	Time Division Multiple Access
TDM	Time-Division Multiplexing
TVS	Transient Voltage-Suppression
TTL	Transistor-Transistor Logic
TCP/IP	Transmission Control Protocol/Internet Protocol
URI	Uniform resource identifier
UART	Universal Asynchronous Receiver/Transmitter
UDDI	Universal Description, Discovery, and Integration
UPnP	Universal Plug and Play
USB	Universal Serial Bus
UNED	Universidad Nacional de Educación a Distancia
UTS	University of Technology, Sydney
USBTMC	USB Test and Measurement Class
USS	User Side Scheduling Service
VME	Versa Module Europe
VHDL	Very-High-Speed Integrated Circuits HDL
VLSI	Very-Large-Scale Integration
VI	Virtual Instrument
VISA	Virtual Instrument Software Architecture
VISIR	Virtual Instrument Systems in Reality
VLE	Virtual Learning Environment
VNC	Virtual Network Computing
VPN	Virtual Private Network
VXI	VME eXtensions for Instrumentation

VCO	Voltage Controlled Oscillator
WDMA	Wavelength-Division Multiple Access
WADL	Web Application Description Language
WoT	Web of Things
OWL	Web Ontology Language
WS-BPEL	Web Services Business Process Execution Language
WSDL	Web Services Description Language
WBT	Web-Based Training
WCF	Windows Communication Foundation
WWW	World Wide Web

This page intentionally left blank.

List of Figures

Figure 1. Remote laboratory for electronic circuits' practices [16]. 52

Figure 2. RemotElectLab [18]. 53

Figure 3. NetLAB [19]. 54

Figure 4. remote laboratory for microelectronic circuit fabrication [19]. 57

Figure 5. Remote laboratory for DC machines [23]. 60

Figure 6. Remote laboratory for DC motor speed control [24]. 61

Figure 7. Remote laboratory for measuring the I-V characteristics of a solar cell [26]. 63

Figure 8. Remote laboratory for Active noise control [27]. 66

Figure 9. Remote laboratory for measuring elastic properties of an aluminum specimen [31]. 69

Figure 10. Remote laboratory for learning control theory with DC motors [33]. 71

Figure 11. Remote laboratory for industrial quadruple-tank [35]. 74

Figure 12. Remote laboratory for programming a microcontroller [37]. 77

Figure 13. Remote laboratory of a LEGO Mindstorms NXT robot [38]. 79

Figure 14. Commercial Multidisciplinary experimentation platforms from NI. 81

Figure 15. A component board with 10 DPST. 92

Figure 16. The common propagating nodes within the relay switching matrix. 92

Figure 17. The internal interconnection of the instrument boards. 94

Figure 18. VISIR installed at UNED. 95

Figure 19. Selecting virtual interfaces of instrument from different models. 98

Figure 20. Simulated work bench of VISIR. 99

Figure 21. A function generator message based on the experiment protocol. 100

Figure 22. The role of IVI standard. 102

Figure 23. VISIR components and overall operation cycle. 104

Figure 24. Video tutorials in UNED-COMA. 107

Figure 25. Reservation system in MOOC-UNED. 109

Figure 26. New designed industrial electronics circuits. 110

Figure 27. Designed circuits connected to the matrix. 115

Figure 28. Online retrieved measurements form the new designed experiments. 116

Figure 29. Class structure of Lab2go ontology [51] 125

Figure 30. Integrating remote laboratories into LMSs. 127

Figure 31. Personal Learning Environment (PLE). 128

Figure 32. Architecture of RLMS. 129

Figure 33. ISA for interactive experiment. 131

Figure 34. Sahara architecture. 134

Figure 35. Communication between two RLMSs using API calls. 138

Figure 36. ReLaSIS illustrative scenario. 139

Figure 37. Service-oriented LMS 156

Figure 38. Mashing up imported services. 160

Figure 39. Laboratory as a Service (LaaS) model. 163

Figure 40. Service description file. 172

Figure 41. Modular remote laboratories architecture. 174

Figure 42. Topology of relay switching circuits. 176

Figure 43. Stand-alone and modular configurations of ATE. 180

Figure 44. Topology of a hybrid bus system. 188

Figure 45. Market share of laboratory server software (chart). 192

Figure 46. MATLAB’s main components for remote laboratories development. 195

Figure 47. LabVIEW’s main components for remote laboratories development. 199

Figure 48. Three typical scenarios illustrating the role of ActiveX and Java applets in the development of remote laboratories. 204

Figure 49. LaaS scenario of Example 1. 207

Figure 50. LaaS scenario of Example 2. 209

Figure 51. Flowchart of the motor-tacho laboratory’s operation cycle. 214

Figure 52. LaaS: motor-tacho laboratory. 215

Figure 53. Motor-tacho laboratory. 216

Figure 54. NI USB-6009 DAQ card pin-outs [102]. 217

Figure 55. Block diagram of the motor-tacho laboratory’s software. 218

Figure 56. Configurations of the DAQ tasks. 219

Figure 57. MATLAB control code written in an .m file. 220

Figure 58. Front panel of the motor-tacho laboratory’s software. 223

Figure 59. Web service implementation in LabVIEW. 225

Figure 60. Block diagram of the motor-tacho laboratory’s software (with Shared Variables). 227

Figure 61. method1.vi. 228

Figure 62. method2.vi. 229

Figure 63. Final LabVIEW project items panel. 232

Figure 64.Topology of Web service implementation in LabVIEW 233

Figure 65. Webcam preview using VideoCapX ActiveX control in LabVIEW. 234

Figure 66. Video streaming using VLC. 235

Figure 67. RESTful Web service response to method1. 239

Figure 68.Streaming of tachometer reading. 240

Figure 69.Response to method2 at the motor.vi . 241

Figure 70. RESTful Web service response to method2. 242

Figure 71. Data written to the database. 243

List of Tables

Table 1. PXI platform installed at UNED. 91

Table 2. Corresponding board number of each I^2C label. 95

Table 3. IVI base and extension capabilities of an oscilloscope. 103

Table 4. Survey on VISIR Deployment at UNED. 106

Table 5. Preliminary survey about the profile of the enrolled students in COMA. 108

Table 6. Curriculum of the RIPLECS master's degree program. 120

Table 7. A comparison between ISA and Sahara. 136

Table 8. Comparison between different Remote laboratories integration scenarios. 142

Table 9.A comparison between SOAP and REST. 172

Table 10. Comparison between different communication buses. 186

Table 11. Market share of laboratory server software. 191

Table 12. List of selected Global Variables. 226

Table 13. URL Mappings of the two HTTP methods, method1 and method2. 231

This page intentionally left blank.

Chapter 1

1. Introduction

Science, Technology, Engineering, and Mathematics (STEM) education is one of the foundations of the developed countries that, in turn, promotes job growth, knowledge, technological advances, and economic prosperity. In a report from the Georgetown University Center on Education and the Workforce [1], engineering jobs are forecasted to occupy the 28% of STEM jobs by 2018 compared with 13% in life and physical science, 6% in architecture and technical fields, and 2% in mathematics, while computer related jobs dominates with over 51%—albeit computer related jobs commonly draw from multiple disciplines. Accordingly, education in engineering is increasingly seen as an important topic with computer application in mind. During the past 100 years, the five major shifts in engineering education were [2]: (1) a shift from hands-on and practical emphasis to engineering science and analytical emphasis; (2) a shift to outcomes-based education and accreditation; (3) a shift to emphasizing engineering design; (4) a shift to applying education, learning, and social behavioral sciences research; and (5) a shift to integrating information, computational, and communications technology in education. The first two shifts have already occurred while the latter three are still in process.

Following the World War II, the fact that many of the great inventions that had occurred were made by individuals educated as scientists rather than engineers caused a rethinking of engineering curricula. At that point, a shift started to occur from training to education by adding more math and science, more synthesis to analysis, and more innovation to book design. This led to the formation of the National Science Foundation (NSF) (www.nsf.gov), established in 1950, that changed the nature of engineering curricula ensuring that research is fully integrated with education so that today's revolutionary work will also be training tomorrow's top scientists and engineers. Later in the late 1990s and the early 2000s, Accreditation Board for Engineering and Technology (ABET) (www.abet.org) developed a wholesale change in accreditation of engineering programs in order to make them outcome based, what was known initially as Engineering Criteria 2000 (EC 2000). Nonetheless, comparing with the extensive evolution of the global economy and worldwide competition in the industrial markets engineering education has been progressing at a very slow pace and it hasn’t witnessed a significant change in the past 100 years in the way that educators instruct students.

Traditionally, engineering education has been content-centered, design-oriented, and permeated by the development of problem solving skills. In a similar fashion, universities have been reluctant to address industry and labor markets specific needs. As a result, engineering graduates have been well prepared technically, but industry has been frustrated owing to the lack of professionals. While strong technical skills will continue to provide the foundation for all engineering sub-disciplines, engineering graduates will need to demonstrate effective skills of communication, creativity, entrepreneurial thinking, and collaborative and team work. Of particular concern is understanding business in a global context and the importance of societal influences on engineering solutions. Evidently, the amount of content deemed necessary for graduates of engineering degree programs has steadily increased and the most competitive engineers will need to embrace a broader professional role to respond to globalization and its challenges.

In a knowledge society, education needs to be re-thought in depth to adapt to this new kind of society characterized by genuinely student-centred learning rather than the fixed traditional learning pattern. In student-centred learning the responsibility is shifted from teachers to students; students are expected to maximize their learning outcomes using relevant technologies and their own competencies. This learning pattern was announced as one of the principle commitments of the bologna process, for the progress of the European Higher Education Area (EHEA) (www.ehea.info), in its last ministerial conference in Bucharest in April 2012—stressing as well on other factors such as learning outcomes, quality assurance, mobility, and evaluation frameworks.

The purpose is to create graduates who can evolve seamlessly into a mode of lifelong learning, continuing education, or even vocational education to cope with such knowledge society [3]. Learning or knowledge acquiring is no longer confined to certain age, place, or time. Instead, it can be seen as something voluntary and self-motivated that takes place throughout life and on an on-going basis from our daily interactions with others and with the world around us.

In 1969, the world's first successful open university, The Open University (OU), was established in United Kingdom, followed by many successful initiatives around the globe that offer higher education on a part-time and/or distance learning basis—including people with health disabilities—such as Spanish University for Distance Education (UNED) in Spain and Latin America, Open Universities Australia (OUA) in Australia, Indira Gandhi National Open University in India, and Arab Open University (AOU) in Middle East and Africa. However, full-distance education programs weren't likely until the evolution of Information and Communication Technologies (ICT) and their application in education, what is now known as E-learning. Oftentimes, it is also referred as Technology-Enhanced Learning (TEL), Computer-Based Training (CBT), Internet-Based Training (IBT), Web-Based Training (WBT), and Online education. E-learning has redefined the concept of distance learning and the delivery of educational resources allowing ubiquitous interactive learning with neither geographical, economic, demographic, nor time constraints. Prior to this, distance education was limited to correspondence courses and televised lectures. The adoption of technologies in education started in 1972 with the introduction of electronic calculators and kept progressing till the advent of highly scaffold and personalized Virtual Learning Environments (VLEs) and online laboratories. Nowadays, entire degree programs and courses may be taken on campus, online, onsite at a company, or in any combination. For example, hybrid learning or Blended Learning (B-Learning) is touted as a means to conserve traditional classroom pattern and simultaneously to allow the convenience facilitated by E-learning. Not only the methods of delivery are being changed, but the relationships among universities, institutions, and corporate entities are now in flux. For example: a student can take online courses, not offered by his/her institution, at another institution; workers at a company can take a course at a university thanks to a prior agreement between the university and their company; a remote expert discussant can be added to an online classroom; or collaborative research teams can be formed across institutions.

In 2002, Massachusetts Institute of Technology (MIT) have sparked the global Open Educational Resources (OER) movement by its MIT OpenCourseWare (OCW) (www.ocw.mit.edu) initiative and after announcing that it was going to putting its entire course catalog online in order to enhance human learning worldwide by the availability of a Web of knowledge. As of November 2011, over 2080 courses were available online. MIT was then quickly followed by the creation of the OCW Consortium (www.ocwconsortium.org) which now unites over 300 institutions, corporates, organizations, and consortia from over 40 countries around the world with materials from thousands of courses accessible. Extending the concepts of OCW, Massive Open Online Courses (MOOCs) was originated in 2008 within the OER movement. MOOCs are open online courses that are more structured formal and aiming at large-scale interactive participation. Only a few percent of the tens of thousands of students who may sign up complete the course. Typically they do not offer academic credit or charge tuition fees but in some cases they offer the possibility of earning academic credit or certificates based on supervised examinations. Subsequently, several providers by elite universities have emerged such as edX (www.edx.org), which was founded by MIT and Harvard University. The rapid expansion of MOOCs has sparked commercial interest from venture capitalists and major corporations who wanted to enter the higher education market. Thereby, new commercial start-ups such as Coursera (www.coursera.org) and Udacity (www.udacity.com) have also been launched in collaboration with prestigious universities, offering online courses either for free, charging or only charging a small fee for the final certification. Some partner universities offer credit for their courses and apply fee to those who want to have some extra assignments and work with an instructor and be assessed. Other open education initiatives have been around such as Udemy (www.udemy.com), P2PU (www.p2pu.org) and Khan Academy (www.khanacademy.org). Different ideologies have driven MOOCs in two different pedagogical directions: the earlier connectivist-based MOOCs (cMOOC), which was based on exploring new pedagogies beyond traditional classroom and emphasizing that learning and knowledge emerge from interaction, creativity, autonomy and informal social networking learning relatively free from institutional constraints; and the later content-based MOOCs (xMOOCs) such as those offered by Coursera and edX, which emphasize a more traditional and behaviorist learning approach through instructional methods with video presentations, short quizzes and testing.

Over the time, the adoption of E-learning in academia and workplace are getting more common thanks to the rapid pace of evolution of ICT. The adoption of E-learning is rapidly evolving and the process is irreversible. According to “The 2011 Survey of Online Learning” of the Sloan Consortium (www.sloanconsortium.org) [4], 31% of higher education students now take at least one course online and 65% of higher education institutions now say that online learning is a critical part of their long-term strategy. The recent noticeable leaps witnessed by E-learning and its high rate of promulgation among universities and institutions have hampered meaningful research in fields related to education and pedagogy such as educational psychology and cognitive science; these fields include, but not limited to, curricula design, instructional methods and design, policy innovations, constructivism, and behaviorism. Obviously, in technology and engineering related fields as well; these fields include, but not limited to, ICT development and transfer, educational platforms, interoperability and integration between systems and components, security, authentication and policies, cloud and federated access, reusability, virtual world, standards and interfaces, Mobile Learning (M-Learning), and virtual and remote laboratories.

1.1. Hands-on Laboratories

Laboratories or hands-on laboratories have long been acknowledged as a key element and an indispensable integral component of engineering education. No one disputes the vital role of experimentation, particularly in engineering and the applied science, in order for students to observe scientific phenomena, consolidate their understanding of theoretical concepts, and thus become expert practitioners of science [5]. It was reported that students retain 25% of what they hear, 45% of what they hear and see, and 70% if they use the “learning-by-doing” methodology [6]. According to the ABET’s general criteria of accrediting 2013-2014 baccalaureate level engineering programs, practical skills are considered among essential student outcome that prepare graduates to attain the program educational objectives. This was obvious in the following outcomes listed in the third criterion: 1) an ability to design and conduct experiments, as well as to analyze and interpret data; and 2) an ability to use the techniques, skills, and modern engineering tools necessary for engineering practice. However, in the literature, little attention has been paid to laboratory activities relative to other pedagogical topics. As an example, a survey [7] was done on the first ten years of published articles of the *Journal of Engineering Education* (from 1993 to 2002) concluded that the keyword "laboratory" was solely mentioned in 6.5% of the articles from 1993 to 1997 and in 5.2% of the articles from 1998 to 2002.

During the early years, after the onset of formal engineering education, the focus was balanced between practice and theory. Over the time, the prevailing academic recognition criteria has shifted away from recognizing contributions to undergraduate education toward recognizing research productivity, which has drawn the attention away from costly, space and scheduling-encumbering, and time-consuming activities such as developing instructional laboratories. Laboratories are generally costly, in terms of time and budget, to develop, acquire, administrate, and maintain. They have extremely low overall utilization rates and limited utility beyond specific courses and not often shared among universities despite the high cost of most equipment. The current inflexible operation of, and constrained access to physical laboratories, in addition to the decreasing budgets and increasing in student numbers have put pressure on universities in the delivery of effective practical laboratory education. This has been possible with the advent of online laboratories and the convenience they provide.

1.2. Online Laboratories

Online laboratories are those laboratories that can be accessed and manipulated online. They are either: virtual or remote laboratories. Virtual laboratories are based on simulation delivered in form of software product or Web-based; it doesn't deal with physical equipment. Simulation is the operation over time of a mathematical modeled system that emulates the behavior of a physical system. Simulation is used in many contexts such training, education, and entertainment. During World War II the “Link Trainer” flight simulator were used to improve safety and shorten training time for over 500,000 pilots saving millions of dollars and many lives. In education, simulation is used for scientific modelling of systems to gain insight into their operation and behavior. It is a very useful tool especially for providing illustrations of phenomena in fields that are not easily visualized, such as electromagnetics, nanotechnology, chemistry and physics, and life science. In engineering, simulation software programs are available, that are used for education and at workplace, such as Simulink for multi-domain dynamic systems; National Instruments (NI) Multisim (www.ni.com/multisim) and PSpice (www.cadence.com) for electric and electronic circuits; NI Ultiboard (www.ni.com/ultiboard) for printed circuit board; CYME (www.cyme.com), PowerFactory (www.digsilent.de), PS CAD (www.hvdc.ca/pscad), and PowerWorld (www.powerworld.com) for electric power system. Early criticisms of computer simulations were that they were rigid, unrealistic, and don’t adequately represent real-world systems and behavior. It is generally agreed that computer simulations today cannot completely replace hands-on laboratories but they might be most effective when they are integrated as an adjunct to hands-on laboratories [5].

Remote laboratories, on the other hand, are based on real physical equipment controlled, monitored, and manipulated through the internet. Remote laboratories have been shown to provide significant benefits compared to traditional hands-on laboratories Examples include optimum and organized utilization, resource sharing between institutions to offset costs, more versatile range of experimentation, mitigation of safety issues, and security and tight constrained access which limits either intentional or unintentional misuse. Remote laboratories appeared in the nineties; the idea was initially proposed in [8] and among the earliest implementations reported in education are [9, 10]. In the late 1990s, the release of the Internet server version (6i) of Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW) (www.ni.com/labview) by NI ignited the developments and the promulgation of remote laboratories among universities around the world.

Advocates of online laboratories outline the conveniences associated with their use in terms of cost mitigation, safety, flexibility, and optimizing utilization rate. Advocates of traditional hands-on laboratories argue that students should be exposed to real environments, which is consistent with the constructivism learning theory—though proponents of remote laboratories might argue that nowadays industrial processes are commonly automated and controlled remotely. In a similar fashion, several questions, debates, and empirical comparative studies on whether remote or virtual format is as effective as the traditional hands-on one have been generated after the advent of online laboratories [11-13]. For example, Corter *et al.* [11] found that online laboratories had an advantage in learning outcomes on simple experiment but not in the complex ones. Lindsay and Good [12] noted an existing substantial bias from students and general preference toward hands-on laboratories, with some potential for the other alternatives to be accepted. They also noted that students involved the virtual laboratory group displayed a lesser grasp of the real context than those in the remote group and hands-on group. In turn, students involved in the remote group emphasized hardware objectives in their minds, while students involved in the virtual group emphasized theoretical objectives.

The general conclusion from these studies [4, 8, 13] is that learning outcomes depends on the exact instructions given to group and the different patterns of work and collaboration regardless of the laboratory format. Thus, scaffolding provided along with the laboratory assignment [14] and appropriate formation of laboratory groups [15] are two major factors for learning outcomes.

1.3. **Motivation**

In the last decade, we have witnessed a significant proliferation of remote laboratories, unconstrained by temporal or geographical considerations, in all fields of electrical engineering education thanks to the exponential revolution of digital technologies. The earlier era of remote laboratories development saw more efforts directed to expanding their application range and dealt with commonplace issues such as security, scheduling, and bandwidth, which eventually, and to a great extent, have been overcome.

Current array of concerns is primarily focused on issues related to remote laboratories delivery format and their pedagogical impact as discussed in Section 1.2. These issues encompass their integration with heterogeneous educational systems and coupling with heterogeneous services and learning objects—instead of being monolithic with fixed design—in order to yield a rich scaffold educational environment and hence better learning experience and outcomes. On the other hand, the goal is to promote sharing resources across institutions and hence more availability and cost offset.

Efforts done addressing these needs managed providing solutions for particular systems rather than being generic and consequently, each institution adopted its own solution, which wouldn't often work with different systems from other institutions. As a result of these limitations, it is still unlikely for two institutions to share their lab resources unless they had previously developed their systems from scratch considering such purpose. Likewise, it is still unlikely for an institution to have all its remote laboratories wrapped in a single system; each of the current developed laboratories usually follows different and exceptional integration approach owing to the missing interoperability factor in their architecture design, the missing of standard design pattern for developers to adhere to, and the unclear learning outcomes.

In response to the above mentioned issues, a thorough study has been conducted on the ongoing approaches and the strengths and drawbacks of each, as well, on the nature of the more likely to be the Web of tomorrow, the learning environment in which remote laboratories should be delivered, and the possible methods of access. This is in order to determine the ideal technique for developing and implementing next generation remote laboratories efficiently taking into consideration the technical and pedagogical concerns. As a result, a novel paradigm, Laboratory as a Service (LaaS), was developed. LaaS is a paradigm for developing modular remote laboratories—based on independent component modules—and implementing them as a set of loosely-coupled services to be consumed with a high level of abstraction and virtualization. LaaS aims to tackle the common concurrent challenges in remote laboratories developing and implementation such as inter-institutional sharing, interoperability with other heterogeneous systems, coupling with heterogeneous services and learning objects, difficulty of developing, and standardization.

1.4. Description

LaaS implies the development of remote laboratories as component modules and these components all together are delivered in form of a set of loosely-coupled services to be consumed by users. LaaS strongly relies well-known industrial and Web standards and middleware technologies for its entire communications and inter-communications in order allow interchangeability seamlessly and programmatically. LaaS follows the Service Oriented Architecture (SOA) in terms of defining the relation between laboratory providers, laboratory consumers, and service broker or repository in which remote laboratories are registered and indexed under metadata standards and ontologies. LaaS merges features from cloud computing—in terms of consuming services on-demand with minimum restrictions and higher virtualization—and features from grid computing—in terms of global distribution. LaaS embraces the WoT in terms of coupling with heterogeneous services and bringing objects to the Web for all spectrum of needs—in either formal or informal contexts.

1.5. Objectives

The overall objectives of LaaS are to:

- Define an organized manner for sharing remote laboratories globally among institutions.
- Allow wrapping remote laboratories in any heterogeneous application container (e.g., widget, applet, or any Web client) independently of the underlying technology adopted in both, as well as, their coupling and mashing up with heterogeneous services (e.g., learning objects) across the Web.
- Facilitate maintenance, reusability, and leveraging legacy equipment.
- Allow interchangeability of components between provider and consumer—seamlessly and programmatically— insofar as consumer could contribute with one or more component instead of the fully-reliance on the provider's equipment and facilities.
- Promote online experimentation and discovery in either every day's formal or informal contexts.

- Set principles for a first global standardized design pattern—for remote laboratories development and implementation—to be followed and adopted by remote laboratories developers.

1.6. Research Methodology

This research has been conducted by several chronological stages and constant factors. The constant factors were:

1) Assisting related national and international conferences, workshops, webinars, and seminars.
2) Consulting the literature and the recent ideas and achievements.
3) Participating in related research projects, consortiums, and special interest groups.
4) Realizing visits to other universities and research groups.
5) Publishing and presenting findings and results in prestigious journals and national and international conferences and receiving feedbacks and opinions.

These factors kept me constantly updated with the new challenges and trends. As well, they helped me to define and refine my research objectives. Alongside with these constant factors the accomplishment of this thesis required six principle chronological stages.

The first stage involved my first experience with a remote laboratory through the installation of the Virtual Instrument Systems in Reality (VISIR) in the department and its implementation official undergraduate courses within the department, as well as in the world first Massive Open Online Courses (MOOC) in electronics that is based on remote laboratory sessions.

Having been acquainted with remote laboratories and their implementation, the second stage expanded this experience and involved a thorough study on the state-of-the-art—including applications, technical issues, integration issues, etc.

After the theoretical study, the third stage involved experimenting with state-of-the-art systems developed by other groups such as iLab and Sahara, and defining the current problems and the upcoming challenges in remote laboratories development and implementation.

In the fourth stage, the premise of this thesis was formed and it was matured during my research visit at UTS.

The fifth stage involved expanding the application range of VISIR and developing a first of its kind remote industrial electronics practices to be delivered within a European inter-institutional master degree program based on experimentation with remote laboratories—as one of the major outputs of this thesis.

The final and sixth stage involved writing this thesis and developing a prototype during my research visit at EPFL.

1.7. Accomplishments and Contributions

In order to achieve the targeted goals of this thesis, I have passed a long and joyful journey of knowledge which involves tasks such as research visits, assisting to scientific conferences and events, and affiliation to scientific consortiums and interest groups. Concurrently, I have been frequently disseminating the obtained results by publishing them in conference papers, journal articles, and book chapters. The total activities and accomplishments realized during the production of this thesis can be summarized as follows:

1.7.1. Participation in Research Projects

- **SOLITE**—*Software Libre de Teleformación*—CYTED-P507AC0526.
  - Funds: Programa Iberoamericano de Ciencia y Tecnología para el Desarrollo (CYTED).
  - Budget: 35.000€.
  - Duration: 01/2008 - 12/2011.
  - URL: http://remo.det.uvigo.es/solite/.

- **mPSS**—*mobile Performance Support for Vocational Education and Training*—142788-2008-BG-LEONARDO-LMP.
  - Funds: Lifelong Learning Programme (Leonardo da Vinci sub-programme).
  - Budget: 324.369€.
  - Duration: 10/2008 - 09/2010.
  - URL: http://mpss.dipseil.net/.
- **sLab**—*Integración de Servicios Abiertos para Laboratorios Remotos y Virtuales Distribuidos*—TIN2008-06083-C03/TSI.
  - Funds: Spanish Ministry of Economy and Competitiveness.
  - Budget: 260.000 €.
  - Duration: 01/2009 - 12 /2011.
  - URL: http://www.ieec.uned.es/Investigacion/sLabs/.
- **e-Madrid**—*Investigación y Desarrollo de Tecnologías para el E-Learning en la Comunidad de Madrid*—S2009/TIC-1650.
  - Funds: Programa I+D en tecnología de la Comunidad de Madrid.
  - Budget: 897.000 €.
  - Duration: 01/2010 - 12 /2013.
  - URL: http://www.emadridnet.org/.
- **BEOL**—*Building an Ecology of Online Laboratories*—Award number#1132813.
  - Funds: National Science Foundation (NSF) Catalyzing New International Collaborations Proposal.
  - Sponsor: Massachusetts Institute of Technology (MIT).
  - Budget: 41.000$.
  - Duration: 09/2011 - 02/2013.
  - URL: http://www.nsf.gov/awardsearch/showAward?AWD_ID=1132813.
- **PAC**—*Performance-centred Adaptive Curriculum for Employment Needs*—517742-LLP-1-2011-1-BG-ERASMUS-ECUE.
  - Funds: Lifelong Learning Programme (Leonardo da Vinci sub-programme).
  - Budget: 310.909 €.
  - Duration: 10/2011 - 09/2013.

  - URL: http://pac.dipseil.net/.
- **RIPLECS**—*Remote-labs access in Internet-based Performance-centred Learning Environment for Curriculum Support*—517836-LLP-1-2011-1-ES-ERASMUS-ESMO.
  - Funds: Lifelong Learning Programme (Erasmus Programme).
  - Budget: 389.661€.
  - Duration: 10/2011 - 09/2013.
  - URL: http://www.ieec.uned.es/Investigacion/sLabs/ .
- **EMTM**— *E-business Mobile Training - use of mobile Performance Support System for acquiring e-business management skills*— 2011-1-PL1-LEO05-19883-ESMO.
  - Funds: Lifelong Learning Programme (Erasmus Programme).
  - Budget: 277.720€.
  - Duration: 01/2012 - 12/2013.
  - URL: http://mtraining.eu/.
- **MUREE**—*Modernising Undergraduate Renewable Energy Education: EU Experience for Jordan*—530332-TEMPUS-1-2012-1-JO-TEMPUS-JPCR.
  - Funds: Tempus Programme.
  - Budget: 1.193.189, 74€.
  - Duration: 10/2012 - 10/2015.
  - URL: http://mapec.ju.edu.jo/Muree/Home.html.
- **Go-Lab**— *Global Online Science Labs for Inquiry Learning at School*—530332-TEMPUS-1-2012-1-JO-TEMPUS-JPCR.
  - Funds: Seventh Framework Programme (FP7).
  - Budget: 9.696.582€.
  - Duration: 11/2012 - 10/2016.
  - URL: http://www.go-lab-project.eu/.

1.7.2. **Research Visits**

- **University of Technology, Sydney (UTS)**—Sydney, Australia.
  - Faculty: Faculty of Engineering and Information Technology (IT).

- Center: Centre for Real-Time Information Networks (CRIN).
- Supervisor: Prof. David Lowe.
- Duration: 03/2012 - 09/2012 (6 months).

- **Swiss Federal Institute of Technology in Lausanne (EPFL)**—Lausanne, Switzerland.
  - Faculty: School of Engineering (STI).
  - Center: Automatic Control Laboratory (LA).
  - Supervisor: Prof. Christophe Salzmann.
  - Duration: 05/2013 - 09/2013 (5 months).

In addition to a short day visit (days) to research groups at the following institutions:

- **Polytechnic University of Madrid (UPM)**—Madrid, Spain.
- **University of Deusto**—Bilbao, Spain.
- **Carinthia University of Applied Sciences (CUAS)**—Villach, Austria.
- **Porto Superior Institute of Engineering (ISEP)**—Porto, Portugal.
- **Open University Netherlands**—Heerlen, Netherlands.
- **Blekinge Institute of Technology**—Karlskrona, Sweden.

1.7.3. Assisted Scientific Events and Conferences

- III Jornadas Internacionales U.P.M. sobre Innovación Educativa y Convergencia Europea (inece’09), November 24-26, 2009, Madrid, Spain.
- IX Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica (TAEE), Abril 13-15, 2010, Madrid, Spain.
- IEEE Global Engineering Education Conference (EDUCON), Abril 14-16, 2010, Madrid, Spain.
- IV Jornadas de Redes de Investigación en Innovación Docente de la UNED, March 22-24, 2011, Madrid, Spain.
- Promotion and Innovation with New Technologies in Engineering Education (FINTDI), Mayo 5-6, 2011, Teruel, Spain.

- International Conference on Remote Engineering and Virtual Instrumentation (REV), June 28-July 1, 2011, Brasov, Romania.
- VII International Conference on Engineering and Computer Education (ICECE), September 25-28, 2011, Guimarães, Portugal.
- 41st Annual Frontiers in Education Conference (FIE), October 12-15, 2011, Rapid City, South Dakota, USA.
- I Jornadas Internacionales de Innovación Docente Universitaria en entornos de Aprendizaje Enriquecidos, September 19-21, 2012, Madrid, Spain.
- Jornadas Red Estatal de Docencia Universitaria (RED-U), Escuela Universitaria de Informática (EUI), Universidad Politécnica de Madrid (UPM), January 31- February 1, 2013, Madrid, Spain.
- IEEE Global Engineering Education Conference (EDUCON), March 13-15, 2013, Berlin, Germany.

1.7.4. **Published Journal Articles**

- M. Tawfik, E. Sancristobal, S. Martin, G. Diaz, J. Peire, and M. Castro, "Expanding the Boundaries of the Classroom: Implementation of Remote Laboratories for Industrial Electronics Disciplines ," *IEEE Industrial Electronics Magazine*, vol. 7, no. 1, p. 9, March 19 2013. (JCR 2012: IF 3.758, Q1-10th in Electrical & Electronic Engineering) DOI: http://dx.doi.org/10.1109/MIE.2012.2206872.
- M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, J. Peire, et al., "Virtual Instrument Systems in Reality (VISIR) for Remote Wiring and Measurement of Electronic Circuits on Breadboard," *IEEE Transactions on Learning Technologies*, vol. 6, no. 1, p. 13, March 12 (First Quarter) 2013. (JCR 2012: IF 0.758, Q3-73th in Computer Science, Interdisciplinary Applications).
  DOI: http://dx.doi.org/10.1109/TLT.2012.20.

- G. Góngora-Gallardo, M. Castro-Gil, A. Colmenar-Santos, and M. Tawfik, “Efficiency factors of solar collectors of parallel plates for water,” *Solar Energy*, vol. 94, p. 9, August, 2013. (JCR 2012: IF 2.952, Q2-21th in Energy & Fuels). DOI: http://dx.doi.org/10.1016/j.solener.2013.05.014.
- R. Gil, E. S. Cristóbal, M. Tawfik, S. Martín, A. Pesquera, G. Díaz, et al., "Applications and Security in the Practical Competitiveness Implementation within Learning Environments," *ARBOR Ciencia, Pensamiento y Cultura*, vol. 187, no. 3, p. 17, December 2011. DOI: http://dx.doi.org/10.3989/arbor.2011.iExtra_3.
- M. Tawfik, E. Sancristobal, S. Martín, C. Gil, A. Pesquera, P. Losada, et al., "VISIR: Experiences and Challenges," *International Journal of Online Engineering (iJOE)*, vol. 8, no. 1, p. 8, February 2012. DOI: http://dx.doi.org/10.3991/ijoe.v8i1.1879.
- S. Martín, E. S. Cristóbal, R. Gil, M. Tawfik, A. Pesquera, P. Losada, et al., "DIEEC (Departamento de Ingeniería Eléctrica, Electrónica y de Control), UNED," *Revista Iberoamericana de Informática Educativa (IE Comunicaciones)*, vol. 15, p. 9, January-June 2012. URL: http://dialnet.unirioja.es/servlet/articulo?codigo=4028850.
- M. Tawfik, E. Sancristobal, S. Martin, R. Gil, A. Pesquera, M. J. Albert, et al., "European Online Master Degree Program for Addressing Labor Market Demands," *International Journal of Online Engineering (iJOE)*, vol. 8, no. 3, p. 7, November 2012. DOI: http://dx.doi.org/10.3991/ijoe.v8iS3.2243
- R. Gil, G. Diaz, M. Tawfik, F. Garcia-Loro, A. Pesquera, E. Sancristobal, et al., "Fingerprint Verification System in Tests in Moodle," *IEEE Journal of Latin-American Learning Technologies*, vol. 8, no. 1, p. 8, February 2013. DOI: http://dx.doi.org/10.1109/RITA.2013.2244702.

1.7.5. **Published Book Chapters**

- M. Tawfik, E. Sancristóbal, S. Martín, C. Gil, A. Pesquera, S. Ros, et al., "Towards a Better Deployment of Remote Laboratories in Undergraduate Engineering Education," in *Using Remote Labs in Education: Two Little Ducks in Remote Experimentation*, J.

G. Zubía and G. R. Alves, Eds., ed Bilbao: University of Deusto, 2011, pp. 387-402. URL: http://www.deusto-publicaciones.es/index.php/main/libro/913. http://www.amazon.co.uk/Using-remote-labs-education-experimentation/dp/8498303354. http://books.google.es/books?id=PQsXw6q5arIC&hl=en&redir_esc=y.

- M. Castro, G. Diaz, E. Sancristobal, A. Pesquera, S. Martin, R. Gil, et al., "E-learning con Competencias Practicas: Integración de Sistema de Gestión de Aprendizaje y Laboratorios," in *Actas del Seminario de Investigación en Tecnologías de la Información Aplicadas a la Educación (SITIAE 2011)*. vol. 22, M. R. Sanchez and J. A. V. Iturbide, Eds., ed Madrid: Universidad Rey Juan Carlos, 2011, pp. 13-29.

1.7.6. **Published Conference Papers**

- M. Castro, M. Llamas, G. Diaz, E. Sancristobal, S. Martin, R. Gil, *et al.*, "Servicios para Plataformas Educativas: Laboratorios y Aplicaciones," presented at the *IX Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica (TAEE)*, Madrid, Spain, 2010.
- M. Tawfik, E. Sancristobal, S. Martín, C. Gil, P. Losada, G. Díaz, *et al.*, "Design of Practical Activities in Electronics Using VISIR Remote Laboratories," presented at the *VII International Conference on Engineering and Computer Education (ICECE)*, Guimarães, Portugal, 2011, pp. 16-22.
- M. Tawfik, E. Sancristobal, S. Martín, R. Gil, P. Losada, A. Pesquera, *et al.*, "Remote Experimentation in the Electronic Engineering Field: VISIR@UNED," presented at the *VII International Conference on Engineering and Computer Education - ICECE '2011 (Workshop "New Trends in Engineering Education")*, Guimarães, Portugal, 2011. URL: http://romeo.det.uvigo.es/guimaraes2011/.
- E. Sancristobal, M. Castro, S. Martin, M. Tawfik, A. Pesquera, R. Gil, *et al.*, "Remote Labs as Learning Services in the Educational Arena," presented at the *IEEE Global Engineering Education Conference (EDUCON)*, Amman, Jordan, 2011, pp. 1189-1194. DOI: http://dx.doi.org/10.1109/EDUCON.2011.5773298.

- M. Tauste, S. Martin, E. Sancristobal, M. Tawfik, J. Peire, and M. Castro, "Implementation of a remote laboratory for practices in FPGAs Programmable Logic Devices," presented at the *1st Experiment@International Conference Remote & Virtual Labs (exp.at'11)*, Lisbon, Portugal, 2011.
- M. Tawfik, E. Sancristobal, S. Martin, C. Gil, A. Pesquera, P. Losada, *et al.*, "VISIR deployment in undergraduate engineering practices," presented at the *Frontiers in Education Conference (FIE) - 1st Global Online Laboratory Consortium (GOLC) Workshop*, Rapid City, South Dakota, USA, 2011, pp. T1A-1 - T1A-7. DOI: 10.1109/GOLC.2011.6086786https://d.docs.live.net/0d8f84609c2d01ba/Thesis/10.1109/GOLC.2011.6086786 & 10.1109/FIE.2011.6143133https://d.docs.live.net/0d8f84609c2d01ba/Thesis/10.1109/FIE.2011.6143133.
- M. Tawfik, E. Sancristobal, S. Martin, C. Gil, P. Losada, G. Diaz, *et al.*, "Remote Laboratories for Electrical & Electronic Subjects in New Engineering Grades " presented at the *Promotion and Innovation with New Technologies in Engineering Education (FINTDI)*, Teruel, Spain, 2011, pp. T1A-1 - T1A-6. DOI: http://dx.doi.org/10.1109/FINTDI.2011.5936416.
- M. Tawfik, E. Sancristobal, S. Martín, C. Gil, A. Pesquera, P. Losada, G. Díaz, J. Peire, and M. Castro, "A New Node in the VISIR Community," presented at the *International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Brasov, Romania, 2011, pp. 16 - 22.
- E. Sancristobal, A. Pesquera, M. Tawfik, M. Latorre, F. Garcia-Loro, R. G. E. Ruiz, *et al.*, "Learning Management Systems and Online Labs within Virtual Worlds," presented at the *International Conference on Remote Engineering and Virtual Instrumentation (REV),* Brasov, Romania, 2011, pp. 78-82.
- M. Tawfik, E. Sancristobal, S. Martin, R. Gil, A. Pesquera, E. Tovar, *et al.*, "Common Multidisciplinary Prototypes of Remote Laboratories in the Educational Curricula of Electrical & Computer Engineering," presented at *the 119th American Society for*

Engineering Education (ASEE) Annual Conference & Exposition, San Antonio, Texas, 2012, pp. AC 2012-3227. URL: http://www.asee.org/public/conferences/8/papers/3227/download.

- E. Sancristobal, A. Pesquera, S. Martin, R. Gil, M. Tawfik, M. Castro, *et al.*, "Challenges of Applying Online Learning Tools in Distance Learning Courses," presented at the *IEEE Global Engineering Education Conference (EDUCON)*, Marrakesh, Morocco, 2012, pp. 758 - 764. DOI: http://dx.doi.org/10.1109/EDUCON.2012.6201133.
- M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, J. Peire, *et al.*, "On the Design of Remote Laboratories," presented at the *IEEE Global Engineering Education Conference (EDUCON)*, Marrakesh, Morocco, 2012, pp. 311-316. DOI: http://dx.doi.org/10.1109/EDUCON.2012.6201065.
- E. Sancristobal, S. Martín, R. Gil, P. Orduña, M. Tawfik, A. Pesquera, *et al.*, "State of Art, Initiatives and New Challenges for Virtual and Remote Labs," presented at the *12th IEEE International Conference on Advanced Learning Technologies (ICALT)*, Rome, Italy, 2012, pp. 714-715.DOI: http://dx.doi.org/10.1109/ICALT.2012.232.
- M. Tawfik, E. S. Cristobal, A. Pesquera, R. Gil, S. Martin, G. Diaz, *et al.*, "Putting Fundamentals of Electronic Circuits Practices Online," presented at the *Technologies Applied to Electronics Teaching (TAEE)*, 2012, pp. 117-121. DOI: http://dx.doi.org/10.1109/TAEE.2012.6235419.
- M. Tawfik, E. Sancristobal, S. Martin, G. Diaz, and M. Castro, "State-of-the-art Remote Laboratories for Industrial Electronics Applications," presented at the *Technologies Applied to Electronics Teaching (TAEE)*, 2012, Vigo, Spain, pp. 359-364. DOI: http://dx.doi.org/10.1109/TAEE.2012.6235465.
- M. Tawfik, E. S. Cristobal, A. Pesquera, R. Gil, S. Martin, G. Diaz, *et al.*, "Shareable Educational Architectures for Remote Laboratories," presented at the *Technologies Applied to Electronics Teaching (TAEE)*, 2012, Vigo, Spain, 2012, pp. 122-127. DOI: http://dx.doi.org/10.1109/TAEE.2012.6235420.

- R. Pastor, D. Sanchez, N. Aliane, R. Hernandez, G. Mariscal, A. Robles-Gomez, *et al.*, "Structured Remote Laboratory Development," presented at the *Technologies Applied to Electronics Teaching (TAEE)*, 2012, Vigo, Spain, 2012, pp. 314-319. DOI: http://dx.doi.org/10.1109/TAEE.2012.6235457.
- N. Mileva, A. Gochev, M. Milev, D. Ekert, R. Messnarz, K. Slaven, *et al.*, "Special Session: Performance-centered Adaptive Curriculum for Employment Needs," presented at the *Technologies Applied to Electronics Teaching (TAEE)*, 2012, Vigo, Spain, 2012, pp. 348-352. DOI: http://dx.doi.org/10.1109/TAEE.2012.6235463.
- M. Tawfik, E. Sancristobal, S. Martin, R. Gil, A. Pesquera, M. J. Albert, *et al.*, "Labor-Oriented Online Master Degree Program," presented at the *International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Bilbao, Spain, 2012, pp. 108-114. DOI: http://dx.doi.org/10.1109/REV.2012.6293115.
- M. Castro, E. Sancristobal, S. Martin, R. Gil, M. Tawfik, A. Pesquera, *et al.*, "One Step Ahead in the Future of Labs: Widgets, Ubiquity and Mobility," presented at the *International Conference on Remote Engineering and Virtual Instrumentation (REV)*-Keynote, Bilbao, Spain, 2012, pp. 1-10. URL: http://rev-conference.org/REV2012/Keynotes.htm.
- A. C. Caminero, S. Ros, A. Robles-Gomez, L. Tobarra, R. Hernandez, R. Pastor, *et al.*, "Work in Progress: Extending a LMS with Social Capabilities: Integrating Moodle into Facebook," presented at the *Frontiers in Education Conference (FIE)*, Seattle, WA, USA, 2012. DOI: http://dx.doi.org/10.1109/FIE.2012.6462316.
- M. Tawfik, D. Lowe, S. Murray, M. d. l. Villefromoy, M. Diponio, E. Sancristobal, *et al.*, "Grid Remote Laboratory Management System: Sahara Reaches Europe," presented at the *International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Sydney, Australia, 2013, pp. 1-9. DOI: http://dx.doi.org/10.1109/REV.2013.6502889.
- E. S. C. Ruiz, M. Tawfik, S. Martín, R. Gil, A. Pesquera, F. Garcia, *et al.*, "Design, Development and Implementation of Remote Laboratories in Distance Electronics,

Control and Computer Subjects," presented at the *International Conference on Remote Engineering and Virtual Instrumentation (REV)*, Sydney, Australia, 2013, pp. 1-9. DIO: http://dx.doi.org/10.1109/REV.2013.6502903

- M. Tawfik, E. Sancristobal, S. Martin, C. Gil, A. Pesquera, F. G. Loro, *et al.*, "Upcoming Challenges in E-Learning & Online Learning Environments," presented at *the Proceedings of Society for Information Technology & Teacher Education International Conference (SITE)*, New Orleans, Louisiana, United States, 2013, p. 6. URL: http://www.editlib.org/p/48259.
- M. Tawfik, S. Monteso, F. G. Loro, E. Sancristobal, F. Mur, G. Diaz, *et al.*, "Design of Electronics Circuits Practices for an Online Master Degree Program Using VISIR," presented at *the IEEE Global Engineering Education Conference (EDUCON)*, Berlin, Germany, 2013, pp. 1222 - 1227.
  DIO: http://dx.doi.org/10.1109/EduCon.2013.6530262
- E. Sancristobal, P. Orduña, M. Tawfik, F. García, O. Dziabenko, D. López-de-Ipiña, *et al.*, "Widget and smart devices. A different approach for online learning scenarios," presented at the *IEEE Global Engineering Education Conference (EDUCON)*, Berlin, Germany, 2013, pp. 808 - 812.
  DIO: http://dx.doi.org/10.1109/EduCon.2013.6530199
- María José Albert Gómez, Clara Pérez Molina, Gabriel Díaz Orueta, Rosario Gil Ortego, Elio San Cristóbal Ruiz, *et al.*, " Proyectos e Investigación: Construcción y Transmisión del Conocimiento a través del b-learning y su Incidencia en la Formación," presented at *Innovación Docente Universitaria en Entornos de Aprendizaje Enriquecidos*, Madrid, Spain, 2013, pp. 17 - 19. URL: http://congresos.uned.es/w3433/archivos_publicos/qweb_paginas/5157/libroactasiidueaeuned.pdf.
- M. Tawfik, E. Sancristobal, S. Martin, R. Gil, A. Pesquera, S. M. Fernandez, *et al.*, "Ubiquitous and Smart Learning Paradigm for Preparing Qualified and Skilled

Engineers," presented at *the 120th ASEE Annual Conference & Exposition*, Atlanta, 2013, ID#6854.
URL: http://www.asee.org/public/conferences/20/papers/6854/view

- C. Perez-Molina, M. J. A. Gomez, R. Gil, G. D. Orueta, E. Sancristobal, S. Martin, et al., "Performance-Centered Adaptive Curriculum for Employment Needs," presented at *the 120th ASEE Annual Conference & Exposition*, Atlanta, 2013, ID #6942.
URL: http://www.asee.org/public/conferences/20/papers/6942/view
- A. C. Caminero, R. Hernández, L. T. S. Ros, A. Robles-Gómez, E. S. Cristóbal, M. Tawfik, et al., "Obtaining university practical competences in engineering by means of virtualization and cloud computing technologies," presented at *the IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE)*, Bali Dynasty Resort, Kuta, Indonesia, 2013, pp. 301-306.
- R. Pastor-Vargas, R. Hernández, L. T. S. Ros, A. Caminero, A. Robles-Gomez, M. Castro, et al., "Remote experimentation in computer science degrees @ UNED," in *XV Simposio Internacional de Tecnologías de la Información y las Comunicaciones en la Educación (SINTICE 2013)*, Madrid, 2013, pp. 147-152.
URL: http://www.e-ucm.es/sintice2013/sintice2013_62_Pastor.pdf
- M. Tawfik, E. Sancristobal, R. Gil, S. Martin, F. Garcia-Loro, S. Monteso, et al., "Remote Laboratories embedded in an Official Engineering Master's Degree Program," presented at *the XV Simposio Internacional de Tecnologías de la Información y las Comunicaciones en la Educación (SINTICE 2013)*, Madrid, 2013, pp. 143-146. URL: http://www.e-ucm.es/sintice2013/sintice2013_61_Tawfik.pdf
- G. Díaz, F. G. Loro, M. Castro, M. Tawfik, E. Sancristobal, and SantiagoMonteso, "Remote electronics lab within a MOOC: design and preliminary results," presented at the Proceedings of *the 2nd Experiment@ International Conference — Online Experimentation (exp.at'13)*, Coimbra, Portugal, 2013, p. 5.
- A. Robles-Gómez, S. Ros, R. Hernández, L. Tobarra, A. C. Caminero, R. Pastor, M. Rodríguez-Artacho, M. Castro, E. SanCristóbal, M. Tawfik, " Towards an Adaptive

System for the Evaluation of Network Services," presented at the Proceedings of *the 43rd Annual Frontiers in Education (FIE) Conference*, Oklahoma, USA, 2013, p. 7.

- R. Pastor, R. Hernández, S. Ros, D. Sánchez, A. Caminero, A. Robles, L. Tobarra, M. Castro, G. Díaz, E. Sancristobal, and M. Tawfik, " Online laboratories as a cloud service developed by students," presented at the Proceedings of *the 43rd Annual Frontiers in Education (FIE) Conference*, Oklahoma, USA, 2013, p. 6.

1.7.7. Other Publications

- M. Tawfik, "Motor de Reluctancia Variable (VRM),"*Boletín Electrónico de la Rama de Estudiantes de la UNED*, Edition XIII, December, 2009, p. 7.
- M. Tawfik, M. Castro, G. Díaz, P. Losada, S. Martín, E. Sancristobal, and C. Gil, "El Proyecto VISIR," *Boletín Electrónico de la Rama de Estudiantes de la UNED*, Edition XV, January, 2011, p. 6.
- M. Castro, E. Tovar, J. Cubillo, S. Martín, E. San Cristóbal, R. Gil, M. Tawfik, G. Díaz, A. Colmenar, and J. Peire, " Tecnología Educativa en la Enseñanza de la Ingeniería: Estudio de su evolución y Sostenibilidad de la misma," *Responsabilidad Corporativa y Sostenibilidad; Cuaderno Red de Cátedras Telefónica; Cátedra Telefónica de la UNED*, 2011, p. 52.

1.7.8. Other Activities and Merits

- IEEE membership since 2009.
- Collaborated in several activities within IEEE Spain Section from 10/2010 till 01/2012.
- Served as treasurer and secretary for the IEEE student branch of UNED from 12/2010 till 10/2013.
- Active member in the following consortiums:
  - Global Online Laboratory Consortium (GOLC).
  - Virtual Instrument Systems in Reality (VISIR).

    - IEEE P1876™ Standard for Networked Smart Learning Objects for Online Laboratories Working Group.
- Obtained the "best student paper award" for the paper:
    - M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, J. Peire, *et al.*, "On the Design of Remote Laboratories," presented at the *IEEE Global Engineering Education Conference (EDUCON)*, Marrakesh, Morocco, 2012, pp. 311-316. DOI: http://dx.doi.org/10.1109/EDUCON.2012.6201065.
- Served as a reviewer in the following:
    - International Journal of Engineering Education (IJEE), Abril 2013, invited by Editor: Dr. Ahmad Ibrahim, http://www.ijee.ie/.
    - IEEE Network - The Magazine of Global Internetworking, May 2013, invited by Guest Editor: Prof. Nei Kato, http://www.comsoc.org/netmag.
    - Computer Applications in Engineering Education – Wiley, July 2013, invited by Executive Editor: Dr. Magdy Iskander, http://onlinelibrary.wiley.com/journal/10.1002/(ISSN)1099-0542
    - IEEE Global Engineering Education Conference (EDUCON), Abril 17-20, 2012, Marrakesh, Morocco.
    - IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), August 20-23, 2012, Hong Kong.
    - IEEE Frontiers in Education Conference (FIE), October 3-6, 2012, Seattle, WA, USA.
    - IEEE Frontiers in Education Conference (FIE), October 23-26, 2012, Oklahoma City, Oklahoma, USA.
    - IEEE Global Engineering Education Conference (EDUCON), Abril 3-5, 2014, Istanbul, Turkey
- Participated in the organization of the following conferences:
    - IX Congreso de Tecnologías Aplicadas a la Enseñanza de la Electrónica (TAEE), Abril 13-15, 2010, Madrid, Spain.
    - IEEE Global Engineering Education Conference (EDUCON), Abril 14-16, 2010, Madrid, Spain.

1.8. Thesis Outline

The rest of the thesis is organized as follows:

- **Chapter 2, Remote Laboratories Applications in Electrical Engineering Education:** This chapter provides a review on remote laboratories applications in electrical engineering education. The chapter addresses the milestones and the most remarkable achievements in each sub-discipline of electrical engineering. The purpose of this study is to gather information about the current remote laboratories development and implementation problems and the limitations associated with different kinds of applications (e.g., low power applications, real-time applications, etc.). On the other hand, this study paves the way for understanding the technical architecture of remote laboratories and defining their common components independently of the kind of applications. The conclusion of this study sets bases for the modular architecture defined in Chapter 5.

- **Chapter 3, Development and Implementation of Remote Electronics Experiments:** This chapter reports on the development and implementation of two sets of electronics experiments using VISIR at UNED. The first set is targeted to the undergraduate engineering curricula and it was deployed in official undergraduate engineering degrees at UNED. As well, in a Massive Open Online Course (MOOC) delivered by UNED to the public. The other, is a set of new advanced remote electronics experiments oriented to postgraduates and apprentices and addresses labor markets and industrial needs in order to diminish the gap between academia and workplace. Industrial-related issues are emphasized in the design in order to allow understanding the behavior of electronics components. This kind of experiments hasn’t been previously reported in the literature either in conventional lab or remote lab arrangement. This combination, as a result, converted VISIR system into a unique training platform of its kind for remote electronics experiments. Afterwards, the system is tested and measurement results retrieved remotely from the mounted circuits are provided as a case study. Finally, the chapter addresses the implementation of these experiments in an inter-institutional European online master degree program in Information and Communication System (ICS).

- **Chapter 4, Integration into Educational Systems:** This chapter provides a qualitative analysis of the current efforts and approaches for integrating remote laboratories with different types of educational systems and also for communicating two different educational systems for the purpose of sharing their integrated remote laboratories. Strengths and weaknesses of each solution are addressed and a general comparison and discussion are elaborated. This study is helpful for determining the ideal solution to build upon, taking into consideration the technical and pedagogical concerns. Thus, the conclusion of this study, as well, sets bases for the proposed LaaS paradigm defined in Chapter 5.

- **Chapter 5, Laboratory as a Service (LaaS):** In this chapter, the proposed LaaS paradigm is described. Foremost, the chapter provides a briefing on the current problems of remote laboratories development and implementation, and how the proposed LaaS paradigm could help tackling these problems. Afterwards, the chapter starts to describe the context of the LaaS paradigm taking into consideration the nature of the more likely to be the Web of tomorrow, the next generation learning environments, and the possible access methods in order to deduce the best scenario of remote laboratories delivery and implementation. Later, the chapter describes the proposed LaaS paradigm and the of modular remote laboratories concept. Finally, two demonstrative examples are provided, along with a brief summary of the proposed concepts, LaaS and modular remote laboratories.

- **Chapter 6, LaaS Prototype:** In this chapter, the proposed theory is applied to the real world by developing the first modular remote prototype—as a Proof-of-Concept. The prototype is delivered as a set of loosely coupled services—according to the LaaS paradigm—to be consumed within any application container independently of the underlying technology adopted in both. The prototype allows interchanging a component module, the database, between provider and consumer using a standard connector. Consumption results are also provided.

- **Chapter 7, Conclusions:** In this chapter a broad conclusion is drawn along with suggestions for future works and a summary of achievements.

Chapter 2

2. Remote Laboratories Applications in Electrical Engineering Education

In this chapter, the most relevant applications from universities all around the world are addressed and categorized in regard to their sub-discipline. Interdisciplinary fields related to electrical engineering such as biomedical engineering and mechatronics and robotics are also addressed. Since computer hardware engineering is derived from several electrical engineering sub-disciplines and computer software engineering are getting everyday more involved in all applications of applied science, both are not listed as a separate sub-discipline.

2.1. Electronics

Fundamentals of electronic circuits' practices are considered as a core module in all the electrical engineering education sub-disciplines. They involve designing, wiring, and measuring circuits. They deal with: passive components such as resistors, capacitors, inductors, and transformers; active components such as diodes, transistors, and thyristors; power supply instruments and function generators; and measuring instruments such as multi-meters and oscilloscopes. Advanced topics for electronics majors encompass power amplifiers, filters, oscillators and timers, voltage regulators, and power supplies. Further advanced topics includes power electronics, optoelectronics, microelectronics, and nanoelectronics.

Among the relevant applications for this kind of practices is [16], in which a remote laboratory was developed for studying the Direct Current (DC) characteristics of different types of diodes (Light-emitting, silicon, and germanium) enabling switching between diodes online, Figure 1.

Another experiment is presented for studying the charging and discharging curves of capacitors enabling switching between different capacitors and resistors online as well.

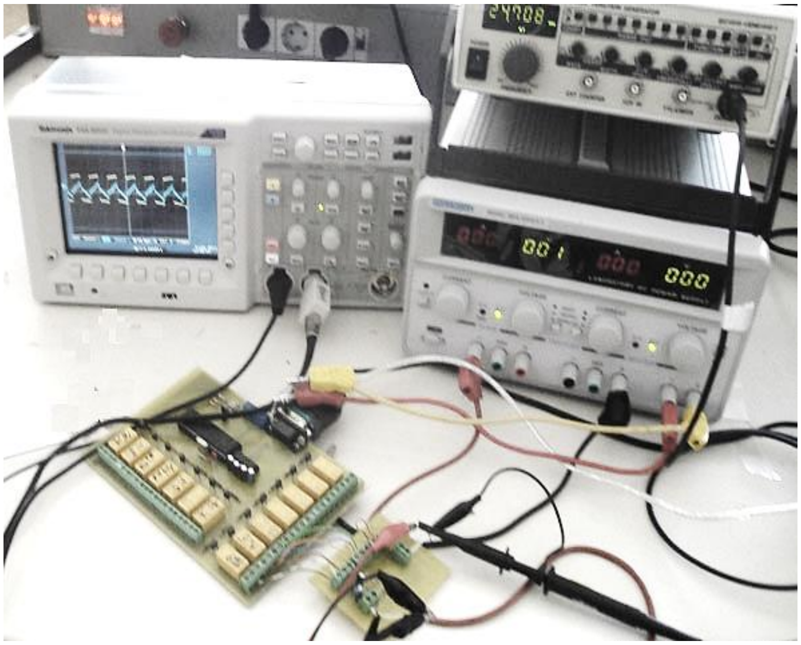


Figure 1. Remote laboratory for electronic circuits' practices [16].

In [17], a remote laboratory was developed for a Bipolar Junction Transistor (BJT) in a common emitter amplifier circuit. It allows studying effect of amplification, adjusting Q-point, changing value of emitter capacitor and load and enables switching between different resistors and capacitors.

In [18], a remote laboratory (known as RemotElectLab) was developed for basic voltage regulator with output current limitation, Figure 2. It allows measuring voltages and currents at several nodes of the circuit and varying components to understand the implications of voltage input and load over voltage regulation, with and without output current limitation. Similar approaches have been reported including experiments on half and full wave rectifiers, clipper and clamping circuits, operational amplifiers, power amplifiers, and filters.

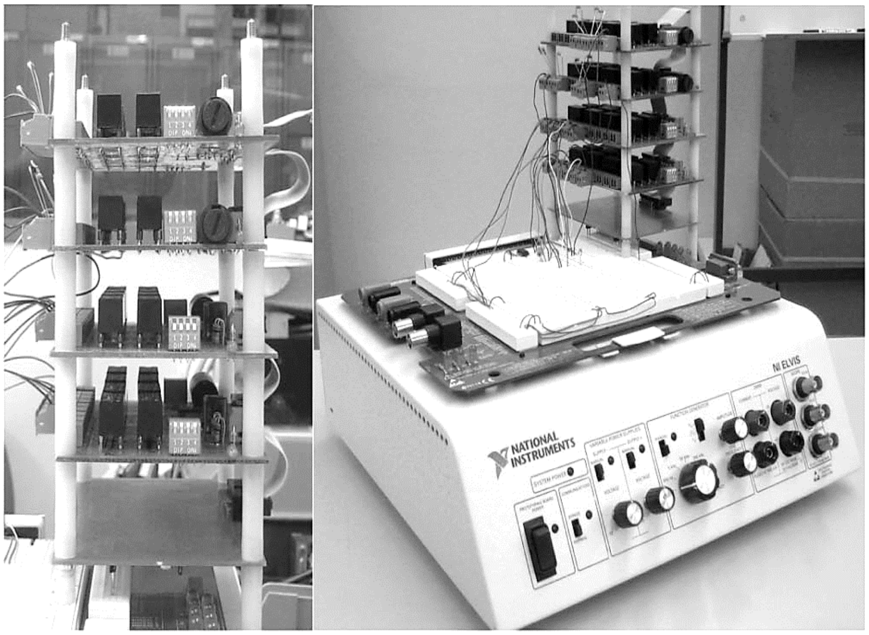


Figure 2. RemotElectLab [18].

2.1.1. NetLab

A remarkable approach, which provides a rich and interactive multiuser collaborative learning environment is NetLab [19], Figure 3. It supports a maximum of 16 two-terminal components and it is primarily used for experiments like investigating RC or RL filters and their transient responses, exploring RLC resonant circuits and other common basic experiments. Its Graphical User Interface (GUI) is its most distinctive part; it has been designed with the intention of giving students the feeling of working in a real laboratory as far as possible. A booking system is also provided within the software application and more than one user are allowed to have full control of the system at the same time. However, the number of concurrent users is limited to three, each with one hour time session, in order to prevent chaos in the laboratory. When the mouse is positioned over a booked seat, the user name and his/her country is displayed. This provides students with the option to choose a laboratory partner even from another country. Lecturer (or administrator) can set a limit on a number of hours per week that each student can book. An unlimited number of users with administrative privileges can access and control the system at any time without booking even if three other users are logged on. A chat window is provided within the GUI; it displays the names of all logged-on users, including administrators. So students can easily detect the presence of a lecturer or another administrator who can help them if they need assistance. A Web cam, fully controllable by the user, is included. A detailed user guide, divided into 8 different chapters, is provided. A control window included in the GUI broadcasts the actions of all users. The available components are: four variable resistors, two variable capacitors, a variable inductor—each with resistance adjustable over 4 decade range, and an in-house built 120:50 turn linear transformer. The available instruments are: a digital oscilloscope, a function generator and a digital multi-meter. All these instruments are also connected to a 16x16 programmable relay switching matrix, which provides the user with an option to wire and configure various electrical circuits from available components and instruments. Other remarkable approach is Virtual Instrument Systems in Reality (VISIR), which is broadly discussed in the next chapter.

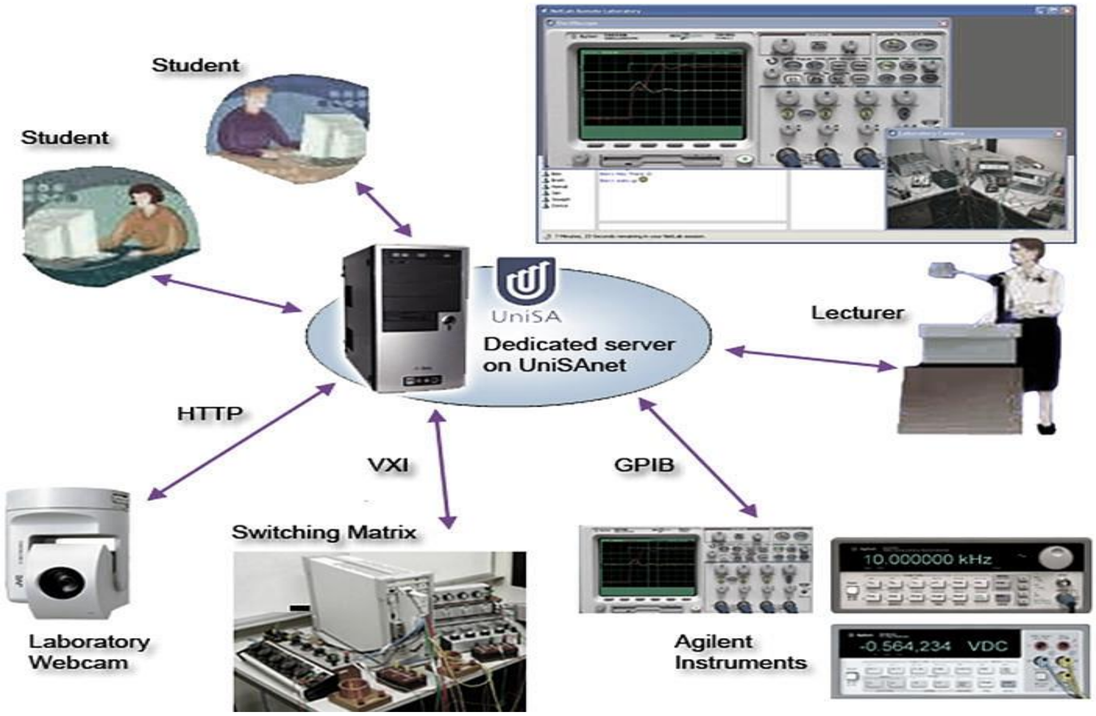


Figure 3. NetLAB [19].

2.2. **Digital Electronics**

Digital electronics fundamentals involves: binary numbers and Boolean algebra; logic gates (e.g., NOT, AND, OR, NAND, NOR, XOR, and XNOR); combinational logic circuits (e.g., adders, comparators, decoders, encoders, Multiplexers (MUXs), and demultiplexers (DEMUXs)); and sequential logic circuits (e.g., flip-flops or latches, counters, and shift registers). Logic gates are implemented using either BJTs as a Transistor-Transistor Logic (TTL) type or Metal–Oxide–Semiconductor Field-Effect Transistors (MOSFETs) as a Complementary Metal–Oxide–Semiconductor (CMOS) logic type, which reduces size and power consumption.

Advanced topics include computer architecture and Programmable Logic Devices (PLDs)—commonly Programmable Array Logic (PAL), Generic Array Logic (GAL), and Complex Programmable Logic Device (CPLD). PALs and GALs are equivalent to a few hundred logic gates while CPLDs are single ICs with multiple PALs or GALs and up to hundreds of thousands of logic gates. PLDs are programmed in Hardware Description Languages (HDLs), mostly popular are Very-High-Speed Integrated Circuits (VHSIC) HDL (VHDL) or Verilog.

A remote laboratory application in digital electronics can be found in [20], where a remote laboratory was developed for programming a MAX II CPLD form Altera (www.altera.com), which is delivered to remote users via Citrix Presentation Server or Citrix XenApp. It allows users to design different digital circuits with VHDL or Altera HDL by means of MAX+PLUS II Software application and upload them to the CPLD. The device is hosted by an evaluation board containing the interface for programming the CPLD and other peripherals like Light-Emitting Diodes (LEDs), push-buttons, Seven-Segment Displays (SSDs), hexadecimal switches, and on-off switches connected to I/O lines of the device. The user can develop a program to interact with these peripherals and test the functionalities and the real behavior of the programmed functions.

2.3. Microelectronics

Microelectronics is a subfield of electronics that relates to the design and fabrication of analog and digital Integrated Circuits (IC) at Large-Scale Integration (LSI) and Very-Large-Scale Integration (VLSI). It deals with micrometer scale or smaller and microelectronic equivalent of electronics components and logic family. This kind of ICs can have from tens of thousands up to several billion of transistors in a single large silicon substrate. CMOS logic is the most popular for constructing LSI and VLSI owing to its energy efficiency. Analog IC design is divided into power IC design and Radio Frequency (RF) IC design. It is used in the design of operational amplifiers, linear regulators, phase locked loops, oscillators and active filters. Digital IC design is used to produce components such as microprocessors, memories and Application-Specific Integrated Circuits (ASICs).

A remote laboratory application example is found in [21] where a remote laboratory, based on the proprietary measurement environment DefSim (www.defsim.com), was developed for studying CMOS physical defects. A dedicated IC with a large variety of intentionally injected physical defects (over 500 different short and open faults) has been implemented in a measurement box, which serves as an interface to the computer and it is connected to it through a Universal Serial Bus (USB) port. DefSim supports two basic test modes: voltage and quiescent supply current (IDDQ) testing, which provides an opportunity for a deeper study of particular defects. Students can work either with all physically realized defects or just Stuck-at Faults (SAF). Laboratory exercise includes: test generation for opens and shorts; circuit truth table calculation; check of the efficiency of SAF test in detection of shorts and opens; detection and localization of an unknown defect; comparison of logic test and IDDQ testing efficiencies; analysis and comparison of test results with results obtained from simulation; and evaluation of different test generation methods in covering real defects.

Another example is found in [22]; a remote laboratory for microelectronic circuit fabrication was developed, Figure 4. It involves the final stage of the fabrication which is the visual inspection and testing of circuits (designed by students in person in previous sessions) on a silicon wafer under a microscope in a cleanroom. Two Webcams are mounted for showing the Device under Test (DUT) allowing remote control of both zoom and focus. Motorized test probes are controlled remotely with high precision and can access any point in the visual field under the microscope. Students can freely move across the wafer surface and explore the characteristics of different components in their design. Lighting is also adjusted remotely. the software is based on the one designed for NetLab [19] and thus it allows booking, animated front panels of instruments, and possibility of student collaboration during remote experiments.

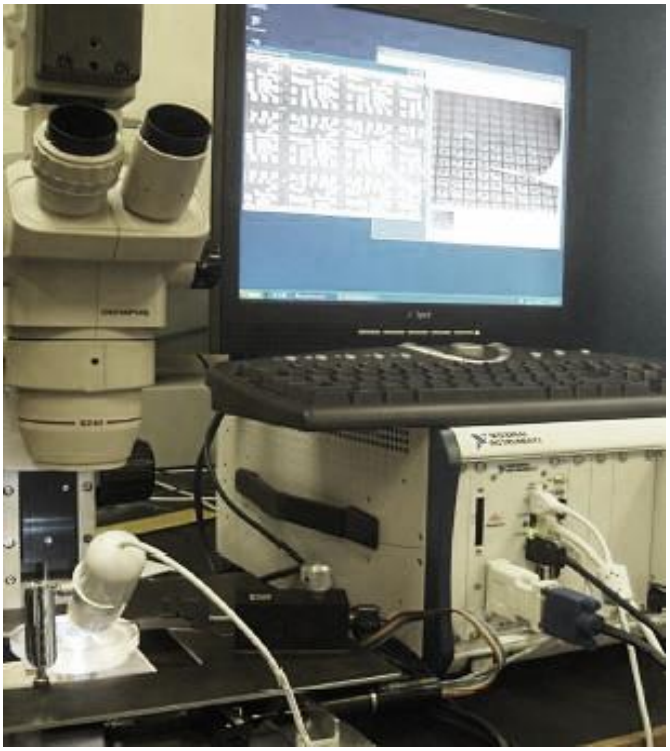


Figure 4. remote laboratory for microelectronic circuit fabrication [19].

2.4. **Power Systems and Electric Machines**

A power system or an electric grid consists of three main subsystems: generation subsystem, transmission subsystem, and distribution subsystem. In the generation subsystem, electricity is most often generated at a power station by electromechanical generators, primarily driven by heat engines fueled by fossil fuels or nuclear fission but also by other means such as renewable energy sources.

The transmission subsystem transmits the electricity to the load centers and it is referred to the transmission substation equipment (e.g., transformers, relays, and circuit breakers) and extra/high overhead/underground transmission lines. The distribution subsystem, which is located near demand centers, continues to transmit the power to the customers and it is associated with low and medium voltage power lines, pole transformers, protection equipment (e.g., circuit breakers and fuses), control equipment (e.g., voltage regulators, capacitors, and relays), and demand side management equipment between the transformer and the customer. The generated electric power is stepped up to a higher voltage at which it connects to the transmission subsystem. The transmission subsystem will move the power long distances until it reaches the distribution subsystem where the power will be stepped down from a transmission level voltage to a distribution level voltage. Finally, upon arrival at the service location, the power is stepped down again from the distribution voltage to the required service voltage.

Energy is usually transmitted within a grid with three-phase Alternating current (AC). High Voltage Direct Current (HVDC) is used for greater efficiency in transmitting large amounts of power over long distances or in submarine power cables. Single-phase AC is used only for distribution to end users and homes, while three-phase AC is used by industry or sites with large poly-phase induction motors. Heavy appliances and industrial machinery use AC power—either in delta (Δ) or in star/wye (Y) arrangement. Modern electronic and digital equipment use DC power and they usually contains an AC/DC converter.

The new emerging grids (smart grids) create an automated and distributed advanced energy delivery network in which the physical infrastructure is replaced with a digital one. Smart grids use ICT to operate in an automated fashion to improve the efficiency, reliability, economics, and sustainability of the production and distribution of electricity.

Electric machine is synonymous with electric motor and electric generator. It is often referred to transformer as well. Electric machines usually have an electronic controller circuit or a drive, which can be a separate inverter or a built-in commutator that allows the machine to be powered by both AC and DC current (e.g., universal motors). Electric machines can be classified into: (1) electromagnetic-rotor machines, those which have some kind of electric current in the rotor to create a magnetic field (i.e., Permanent Magnet (PM) machines, brushed machines, and induction machines); (2) reluctance machines, those which have no windings in rotor but only a ferromagnetic material; and (3) other non-magnetic types such as electrostatic and piezoelectric machines. Electric machines can be: synchronous, meaning that the magnetic field set up by the stator coils rotates with the same speed as the rotor; or asynchronous, meaning that there is a speed difference. PM machines and reluctance machines are always synchronous. Induction machines are usually asynchronous unless there are superconductors in the rotor windings. Brushed machines with rotor windings can be either synchronous or asynchronous depending on the frequency of the DC or AC current feeding the rotor. Selection between different types of electric machines is based on many factors among them efficiency, cost, power, duty cycle, peak torque and controller circuit.

Remote laboratories applications in this field are numerous. In [23], a remote laboratory was developed for controlling and measuring parameters of separately excited dc motor and generator Figure 5. The DC machines are operated by a (0–230 V, 50 Hz) power line supply with a current range of 0–10 A and speed range of 0–1500 rpm. Four Boolean control knobs (i.e., armature control, field control, no load, and load) are available and a graphical plot diagram is generated for each control execution. Two basic experiments are supported for the separately excited dc motor: variation of motor speed with the armature voltage keeping field current constant, and variation of motor speed with field current keeping armature voltage constant. Similarly, two basic experiments are supported for the separately excited dc generator: variation of terminal voltage with generator field current keeping speed constant—at no-load, and variation of terminal voltage with load current keeping generator field current constant—at load. Experimental parameters such as shaft torque, motor output power, generated voltage, and generator output power are calculated using mathematical formulas. A Webcam is provided for audio and video streaming and a safety control is implemented on the machines' speed to avoid their damage.

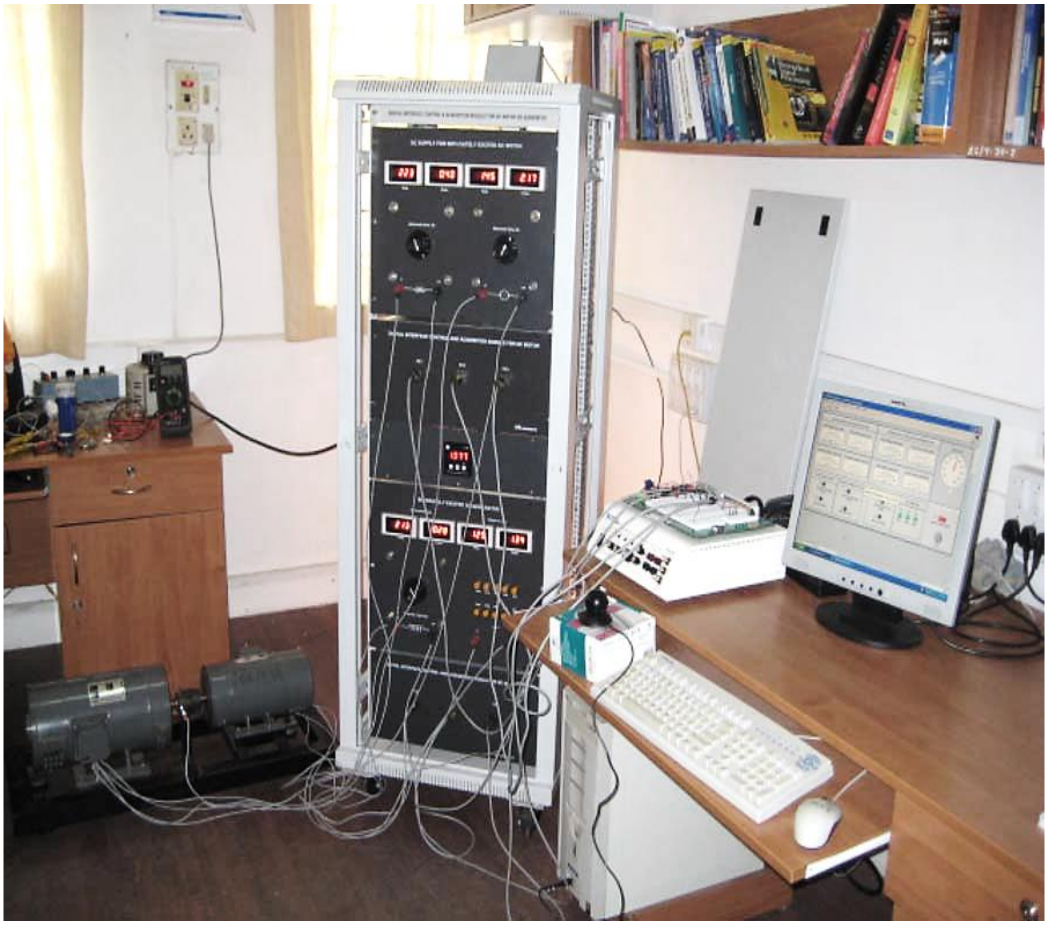


Figure 5. Remote laboratory for DC machines [23].

In [24], a remote laboratory was developed for DC motor speed control using four-quadrant controller unit, Figure 6. The speed control is implemented by changing armature voltage. Students send remotely Pulse Width Modulation (PWM) and direction signals to the motor, which allows it to run in the four quadrants. The PWM signals, the motor current, the voltage, and the speed values are measured and plotted graphs are generated. A Webcam is available for real-time monitoring.

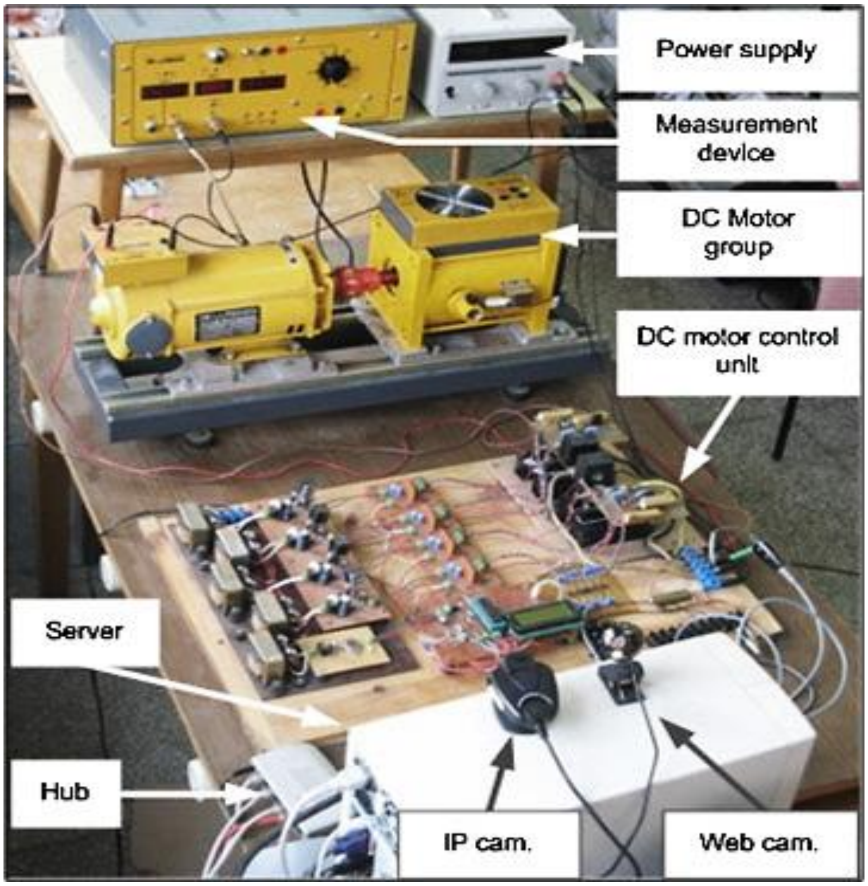


Figure 6. Remote laboratory for DC motor speed control [24].

Another example is found in [25] where remote laboratory was developed for vector-controlled or field oriented-controlled induction motor using high-performance drive—highly optimized for fast arithmetic and lower sampling time (i.e., 100 µs). The vector control permits independent control of the torque and flux producing components of the stator current and thus controlling the speed and/or position of the motor.

2.5. **Renewable Energy**

Renewable energy is a green or eco-friendly source of energy alternative to other energy sources, such as fossil and nuclear fuels derivatives, which result in significant energy security and economic benefits in the short and long term. They come from natural resources which are constantly replenished such as sunlight, wind, rain, tides, plant growth, waves and geothermal heat. The common renewable energy types are: solar (e.g., solar thermal, solar photovoltaic (PV), and artificial photosynthesis); wind, hydro (e.g., tidal and wave), geothermal, and biofuel (i.e., from biomass, solid biomass, liquid fuels, and other biogases). The major concern of engineers and scientists in this field is to provide an efficient and sustainable renewable energy ensuring the matching between the renewal and consumption rate and ensuring the addressing of the global demand, which is expected to be achieved in the near future.

A remote laboratory application is found in [26], wherein a remote laboratory was developed for measuring the I-V characteristics of a 15 cm2 monocrystalline silicon solar cell, Figure 7. A variable capacitor load is applied to the output of the solar cell. A 50 W halogen light bulb with an aluminum reflector—assuring a light spot with a high uniformity—is used as a light source. A temperature sensor with a small thermal inertia is placed in intimate contact with the solar cell. Four photoresistors are placed in the corners of the solar cell in order to obtain the feedback about the lighting level. The equipment is placed inside a box to ensure to eliminate the interference of any external light source. The application allows studying the dynamic behavior of the solar cell, either by switching on/off the light source, modifying the load, or varying the distance between the solar cell and the light source in order to obtain a variable lighting in the range of 50-2000 W/m2. The sample rate and number of samples taken during measurements can be configured. Students can monitor the temperature level, the lighting level. For each adjustment, the graph of the current, voltage, temperature, and spectrum evolution is plotted.

2.6. Signal Processing

Signal processing is a mathematically oriented field, which deals with signal extraction from physical systems along with any subsequent transformation of its attributes, using mathematical and experimental methods, for the purpose of monitoring, analyzing, synthesizing, or manipulation of such information. A signal is a function of time—such as temperature, velocity, blood pressure, video, audio, and image—that could be continues signal (analog signal) or discrete-valued (quantized) discrete-time (sampled) signal (digital signal). Analog signal processing—as in legacy radio, telephone, radar, and television systems—may involve the amplification and filtering of audio signals for audio equipment or the modulation and demodulation of signals for telecommunications. Analog values are typically represented as a voltage, electric current, or electric charge. Analog processing elements include capacitors, resistors, inductors and transistors. Digital Signal Processing (DSP) may involve quality improvement (e.g., noise reduction, image enhancement, and echo cancellation), signal compression (audio compression, image compression, and video compression), or feature extraction (e.g., image understanding and speech recognition) of digitally sampled signals.

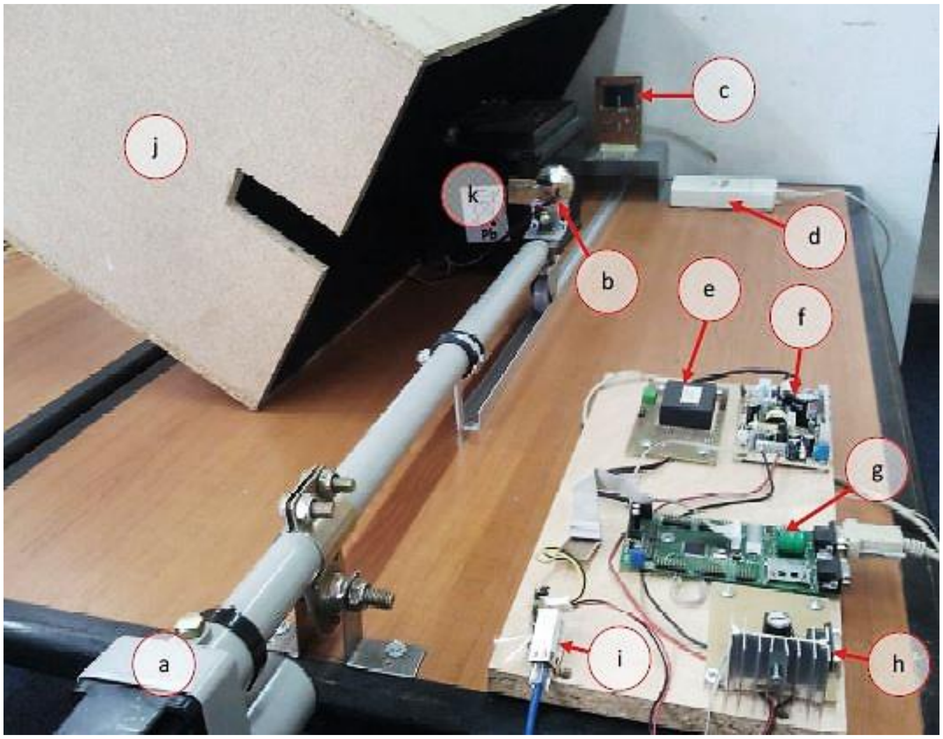


Figure 7. Remote laboratory for measuring the I-V characteristics of a solar cell [26].

DSP entails four common steps: (1) converting physical parameters to electrical signals using sensors; (2) converting analog sensor signals to digital values using Analog to Digital (A/D) converters; (3) applying DSP techniques on the digital signal; and (4) converting the processed digital signals to analog values using Digital to Analog (D/A) converters. Some of the common types of sensors used for collecting data are: microphones, which measure acoustic or sound data; seismometers, which measure earth motion; photocells, which measure light intensity; thermistors, which measure temperature; and oscilloscopes, which measure voltage. DSP is done by digital circuits especially Digital Signal Processors (DSPs). DSPs are specialized type of microprocessors that tend to provide lower-cost solution, with better performance, and lower latency. they must work in real time by processing information as it happens, as in case of such as mobile phones and Personal Digital Assistants (PDAs), and keep pace with the sampling rate of the A/D Converter doing its calculation as fast as the sampled data received.

In [27], a remote laboratory was developed for conducting Active Noise Control (ANC) covering beginners level such as simple experimental measurements to advanced users and even researchers such as algorithm development and their performance evaluation on DSP, Figure 8. Necessary steps involved in an ANC experiment such as validity of ANC, forward path estimation, and active control applied to a broad band random noise (0- 200 Hz) in a ventilation duct. The initial laboratory setup is aimed at a single channel feed forward ANC applied to a circular ventilation duct. Inside the duct are four reference microphones and one error microphone of ordinary quality fixed. Two loudspeakers are placed at each end of the ventilation duct: one acting as a source (primary noise) speaker and the other as a control (secondary noise) speaker. A four channel dynamic signal analyzer is used for analysis of the control and measurement signals and as well acts as a signal source to the primary noise speaker. Adaptive control Least Mean Square (LMS) algorithms is implemented on a DSP board to keep track of non-stationary behavior of the primary noise and steers the secondary speaker. The controller, which may be a Finite Impulse Response (FIR) or an Infinite Impulse Response (IIR) filter, is supplied with some prior information of the noise, known as a reference signal. A filter/amplifier module is used for signal conditioning. The ANC laboratory provides two interfaces named “measurement and configuration client” and “Web-based development environment”. When the laboratory is accessed only the “measurement and configuration client” is presented to the user and the ANC system configuration tab is used to connect the desired microphones in the duct to a DSP channel according to the requirements of experiment at hand. The necessary amplification and filtering (anti-aliasing and re-construction) of the signals from the microphones and control signals from the DSP can be easily adjusted. A GUI of the frontal panel of the signal analyzer is provided for controlling and monitoring the signals. The "Web-based development environment" is launched from the "Web-based development environment" when needed. It steers the DSP board from a client computer over the internet and provides user with functionalities such as file operations project management, compiling source code, downloading the executable to the DSP, and a series of debugging features remotely. The system also allows general experiments in the field of acoustics and digital signal processor. The microphones and speakers installed at different positions along the length of the duct provide the possibility of the ANC laboratory setup to be used for a wide range of acoustic experiments. Based on the dimensions of the duct students can verify and understand plane wave propagation, mode shapes and standing waves produced in the duct. On the other hand, the digital signal processor used in the remote laboratory is a general purpose floating point processor with fixed point processing modules on board as well. It can be used for students to train on DSP programming, optimization, and memory management in general.

2.7. Telecommunications

Telecommunication is a field of engineering that deals with transmission of information through electromagnetic waves. Telecommunication technologies include telegraph, telephone, radio, signal lamps (i.e., using color codes or flashed in a Morse code sequence), television, radar, satellite, computer networks and Internet, and mobile phone. A communications network is a collection of transmitters, receivers, and communication channels that convey an information signal from one/several transmitters to one/several receivers. Communications signals can be either by analog signals or digital signals (where information is encoded as a set of discrete values). Examples of channels include solid, liquid, or gas for sound communications and twisted pair wires (e.g., coaxial cables), optical media (e.g., optical fiber), and free space (vacuum) for communications for non-ionizing electromagnetic waves (visible, infrared, ultraviolet, and radio waves). Telecommunication is either point-to-point (i.e., between one transmitter and one receiver) or broadcast (i.e., between one transmitter and numerous receivers).

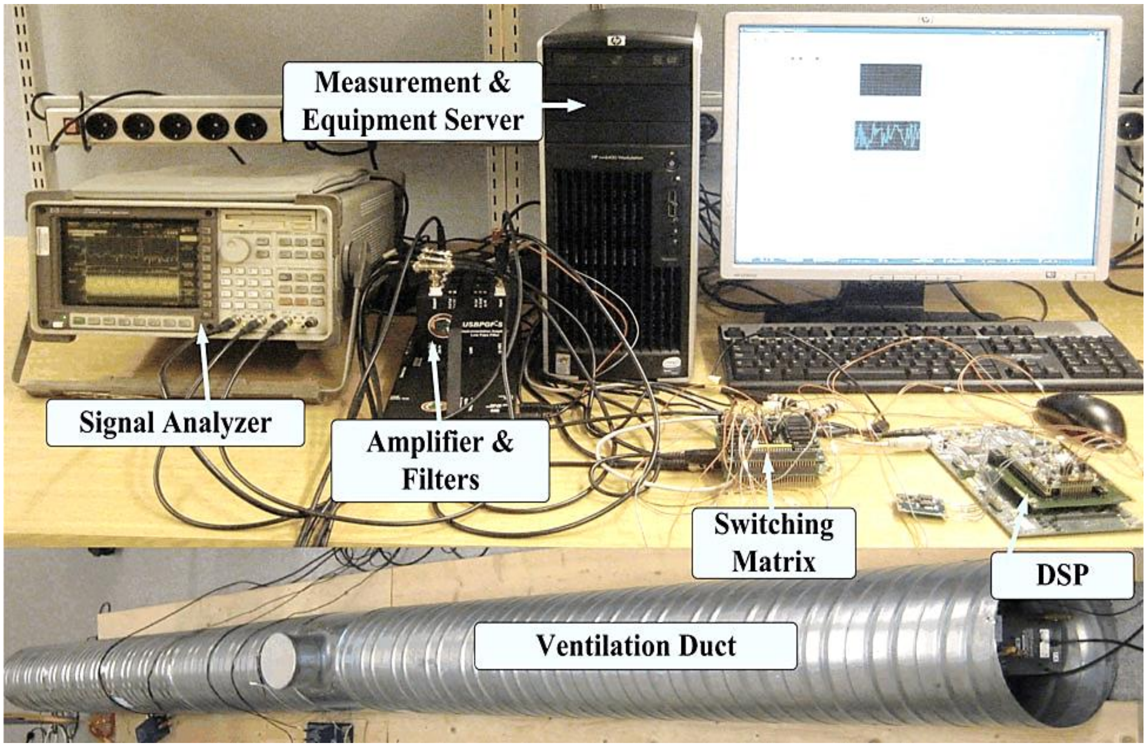


Figure 8. Remote laboratory for Active noise control [27].

Modulation is used to convey digital or analog information signal inside a carrier signal (i.e., over an analog bandpass channel) that can be physically transmitted. Common analog modulation techniques are Amplitude Modulation (AM) and Phase Modulation (PM) Frequency Modulation (FM) and common digital modulation techniques are Phase-Shift Keying (PSK), Frequency-Shift Keying (FSK), Amplitude-Shift Keying (ASK), and Quadrature Amplitude Modulation (QAM). Multiple analog or digital message signals are combined into one signal over a shared medium by means of MUX. Common multiplexing technics are Space-Division Multiplexing (SDM), Frequency-Division Multiplexing (FDM), Time-Division Multiplexing (TDM), Code Division Multiplexing (CDM), and Polarization Multiple-Input Multiple-Output (MIMO) Communications.

A remote laboratory for FM generation circuit with a Voltage Controlled Oscillator (VCO) monolithic integrated circuit ICL8038 is found in in [28]. It enables changing the frequency of the precision waveform generator (proportional to an input DC voltage), the carrier frequency of the FM signal (by switching between different resistors), and the amplitude and frequency of the input signal. The waveforms and the spectrum of the FM signals are monitored and stored for computing experimental parameters such as modulation index and transmission bandwidth of the output signal.

In [29], a remote laboratory was developed to study the effect of the Direct-Sequence Spread Spectrum (DSSS) technique on information signals. It supports simulation and data acquisition modes enabling users to compare real and ideal DSSS communication scenarios. It involves three key stages: generation & modulation of information signal, DSSS signal transmission, and demodulation and recovery of information signal. It allows remote controls such as adjusting information signal type form, amplitude and frequency, varying noise amplitude and length of the modulating Pseudorandom Number (PN) sequence, adding different signal jamming types of various powers, varying receiver's demodulating PN sequence, and adjustment of the tunable low-pass filter. The DSS signal, the information signal, and the recovered signal are graphically displayed and analyzed in time and frequency domain to study the parameters' variation effect in each stage.

Another example is shown in [30] wherein a remote laboratory was developed for realizing and configuring an ordinary AM circuit, in which a DC offset is added to the message signal, giving the liberty of manipulating knops and switches and remote wiring between nodes. The message signal, the carrier signal, and the desired output AM signal are monitored online.

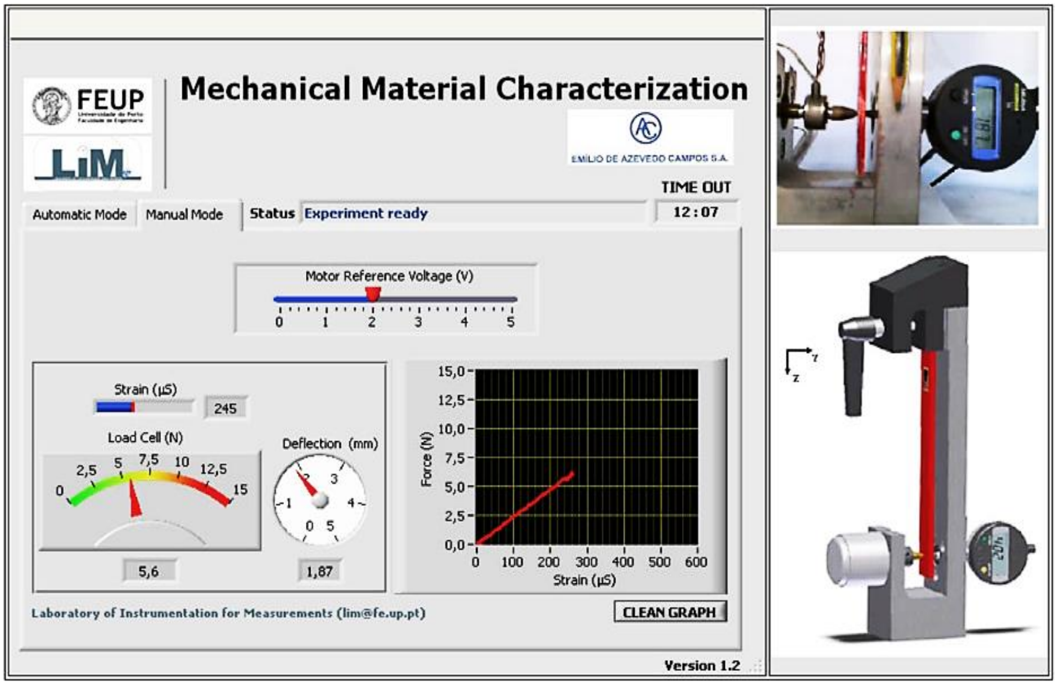
2.8. Instrumentation & Measurements

Instrumentation engineering deals with the design of devices to monitoring, measurement and control of process variables or physical quantities. The basic physical quantities of an instrumentation application are: current, energy, force, frequency, length, mass, pressure, power, resistance, temperature, velocity, and voltage. Instrumentation system usually includes transducers (i.e., sensors or actuators), signal conditioning, signal processing, and display. Sensors convert a signal from one form of energy to another (i.e., commonly to an electrical signal) to allow required conditioning and processing steps to be completed. Signal conditioning consists of amplification, filtering, limiting, and other operations that prepare the raw instrument input signal for further operations. Signal processing applies some algorithm to the input signal in order to obtain meaningful information. Signal conditioning and processing operations may be performed using analog or digital circuit techniques, or using a combination of methods.

Measurements are most commonly made in the International System of Units (SI) system, it was built around seven base units (i.e., kilogram, metre, candela, second, ampere, kelvin, and mole), 22 named and an indeterminate number of unnamed coherent derived units, and a set of prefixes that act as decimal-based multipliers.

A remote laboratory application for measuring elastic properties of an aluminum specimen is found in [31]. It consists of a remotely manipulated cantilever beam instrumented with resistance strain gauges, Figure 9. Students gets numerical, graphical, and live video output information and receive e-mailed experimental results. Several testes are performed by applying a set of discrete loads increments. After completing the stabilization stage for each load increment, the corresponding strain and stress values are digitally recorded and the deformed cantilever shape is graphically presented, which is useful for the determination of Young's modulus and Poisson's ratio.

In [32], a remote laboratory was developed for performing characterization experiments on transducers in general and acceleration and Speed transducers (i.e., accelerometer and tachogenerator) in particular. For the accelerometer's case, the objective is to determine the output voltage of accelerometer with changes in acceleration. For the tachogenerator's case, the objective is to determine the output voltage with changes in speed. Signal conditioners are added to each sensor to amplify and/or filter generated signals. The student set the speed at which the transducer's output is to be recorded. The readings of the angular velocity (i.e., from which acceleration is computed), acceleration, and corresponding output voltage (i.e., from the single conditioner's outputs) are processed and recorded to plot the characteristic curves.



Mechanical Material Characterization

Status Experiment ready

TIME OUT 12:07

Strain (µS) 245

Load Cell (N) 5,6

Deflection (mm) 1,87

Laboratory of Instrumentation for Measurements (lim@fe.up.pt)

Version 1.2

Figure 9. Remote laboratory for measuring elastic properties of an aluminum specimen [31].

2.9. **Control**

Control engineers apply control theory to model dynamic systems (e.g. mechanical, electrical, fluid, etc.) and design controllers, using mathematical modeling, which cause these systems to behave in the desired manner. In a closed-loop control, sensors measure the states or outputs of the dynamic system or the device being controlled and fed them back using actuators as an input to the process and thus making corrections toward the desired performance. In contrast, in open-loop control, no measurement of the system output is used to alter the control—predicting the necessarily outputs to achieve the desired states and assuming there are no disturbances to the system. If the designed control of a device performs without direct human intervention for correction it is called automatic control (i.e., a thermostat is an example of an automatic control device).

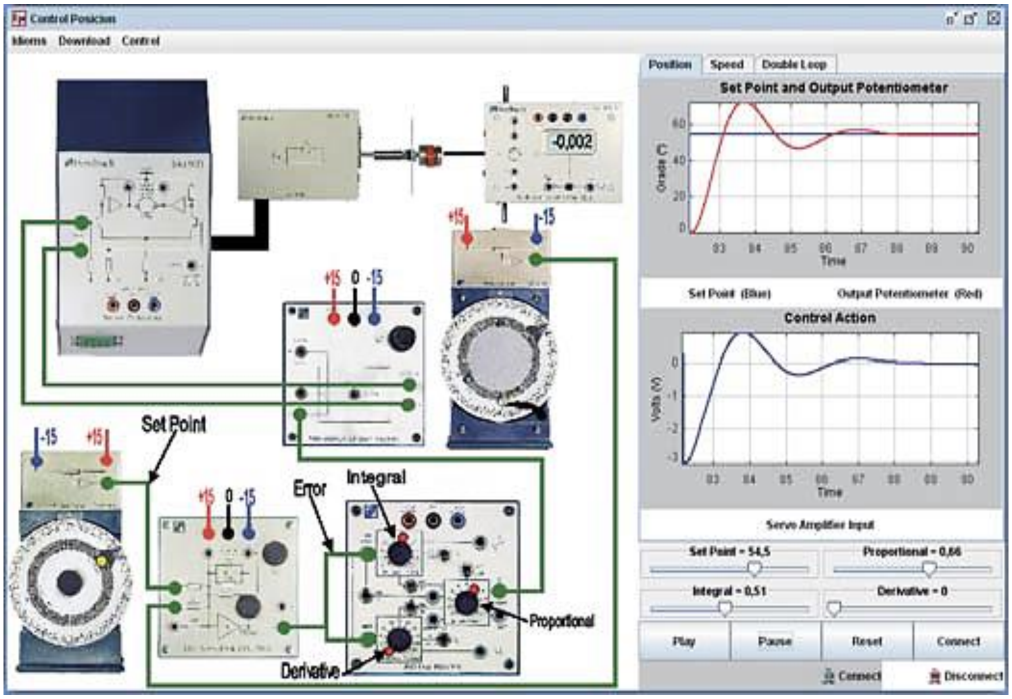
Classical (or conventional) control is limited to Single-Input and Single-Output (SISO) system design, which is analyzed in the time domain using differential equations, in the complex s-domain using Laplace transform, or in the frequency domain by transforming from the complex s-domain. In digital control, Laplace transform is replaced with Z-transform. Proportional Integral Derivative (PID) controllers are the most common controllers designed using classical control. Modern control deals with more sophisticated and MIMO systems. It utilizes the time-domain state space representation which is valid for linear and nonlinear systems.

Modern control methods encompasses nonlinear, adaptive, and robust control. Unlike the frequency domain approach, the use of the state space representation is not limited to systems with linear components and zero initial conditions. For linear systems, stability can be obtained by directly placing the poles of the transfer function using design techniques such as root locus, Bode plot, Nyquist stability criterion, and Full State Feedback (FSF). For nonlinear systems, specific techniques are adopted to linearize the system and to ensure stability such as Gain scheduling, feedback linearization, and Lyapunov's theory. Intelligent control is an emergent alternative control type that uses various artificial intelligence approaches such as fuzzy logic and neural networks.

In [33], a remote laboratory was developed for learning control theory with DC motors and to compare between different control loops, Figure 10. As a hardware, the MS150 DC Complete Modular Servo System from Feedback (www.feedback-instruments.com) is adopted, which is widely used on introductory courses of automatic control. Three control loops are configured in order to study the corresponding feedback effects: 1) position control of a DC motor, using a potentiometer attached to the motor shaft as a sensor in the feedback loop and an ideal standard algorithm PID module as a control algorithm; (2) speed control of a DC motor, using a tachogenerator attached to the motor shaft as a sensor in the feedback loop and also the PID module as a control algorithm; and (3) double feedback loop (position and speed) to improve the position control of the DC motor. In the GUI, a diagram of the overall connection is provided using embedded images of the real modules of the MS150. Optionally, a Webcam is provided to capture the feedback. There is a panel container that comprises two charts for each of the three tabs/options (i.e., position, speed, and double). One of the charts displays the setpoint and the controlled variable (i.e., speed, position, or both) and the other plots the control action computed by the PID algorithm. The values of setpoint and PID parameters can be modified using the sliders.

2.10. **Industrial Control Systems (ICSs)**

ICSs are computer-controlled systems that monitor and control industrial processes, enabling automation and lights-out manufacturing. The incentive for applying ICSs either in infrastructure or facility-based processes is to realize tasks beyond those possible with current human labor capabilities. ICSs reduces costs and operation time or cycle time and replace humans in tasks that involve hard physical, monotonous, or risky work. This increases productivity, quality, accuracy, and robustness. ICSs encompass Distributed Control Systems (DCSs), Supervisory Control and Data Acquisition (SCADA) systems, and Programmable Logic Controllers (PLCs).



Set Point = 54,5

Proportional = 0,66

Integral = 0,51

Derivative = 0

Figure 10. Remote laboratory for learning control theory with DC motors [33].

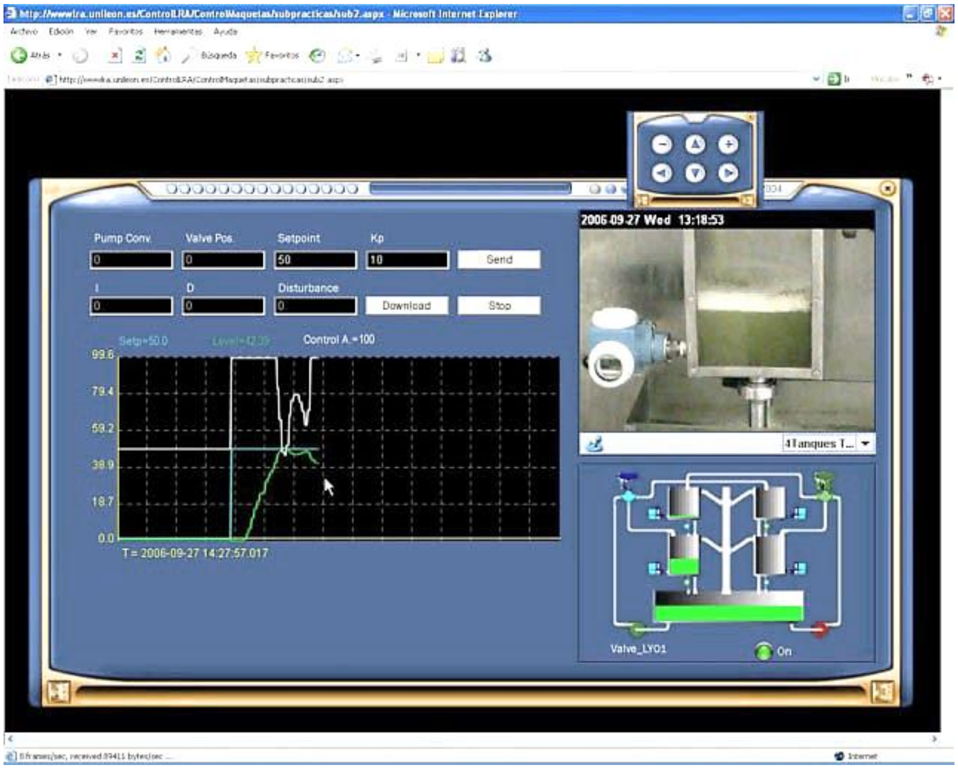
DCSs refers to a control systems in which processing is performed at the location most beneficial to the overall system goals; information flows throughout the system and is available wherever and whenever it is needed in a standard format which is unambiguous to the receiver. SCADA systems exhibit predominantly large-scale distributed control system that can include multiple sites, and large distances. The differences between a DCS and a SCADA is often subtle and boundaries between both systems are blurring as time goes on, owing to advances in technology that allow the functionality of each to overlap. A DCS or a SCADA usually consists of the following subsystems: (1) Human–Machine Interfaces (HMIs) through which the human operator controls and monitors the process; (2) a Master Terminal Unit (MTU); (3) Remote Terminal Units (RTUs); and network infrastructure for data communication between the MTU and RTUs. MTU is the master supervisory computer or central host that might be a single computer or a cluster of computer servers. A RTU is a stand-alone data acquisition and control units that interfaces objects in the physical world to a DCS or a SCADA system. In modern RTUs, most control actions are performed automatically by RTUs and MTU functions are usually restricted to basic supervisory level intervention.

A PLC is an industrial microcontroller system, armored for severe industrial environments conditions, with hardware and software that are specifically adapted to the industrial environment. It is programmed in ladder logic, which is conceived to be easy to grasp by technicians. Typically a ladder logic languages from a manufacturer will not be completely compatible with products from other manufacturer. The International Electrotechnical Commission (IEC) 61131-3 specification has helped to standardize programming PLCs defining graphical and textual PLC programming language standards: Ladder diagram (LD), Function Block Diagram (FBD), Structured Text (ST), Instruction List (IL), and Sequential Function Chart (SFC). Modern PLCs can be programmed in other variety of ways such as specially adapted dialects of BASIC and C.

Typical legacy ICSs communication protocols include Local Operating Network (LON), Profibus, Controller Area Network (CAN), INTERBUS, Modbus, and RP-570. These communication protocols are all ICS-vendor specific but are widely adopted and used. Standard protocols recognized by all major ICS vendors are IEC 60870-5-101/104, IEC 61850, OLE for Process Control (OPC) Unified Architecture (UA), and Distributed Network Protocol (DNP) 3. Heavy use of Commercial Off-The-Shelf (COTS) technology and protocols such as Transmission Control Protocol (TCP)/Internet Protocol (IP) blurred the line between traditional and industrial networking and at the same time, made industrial network systems vulnerable to threats that affect traditional networks. Vendors of ICSs and control products have begun to address the risks by developing specialized industrial firewall and Virtual Private Network (VPN) solutions for TCP/IP-based ICS networks.

In [34], a remote laboratory was developed to allow students programming a PLC that derives several experimental stations. Students solve a number of exercises proposed by the instructor using various IEC 61131–3 languages— commonly SFC and ladder logic. Students write their code in the Integrated Development Environment (IDE) CODESYS (www.codesys.com) and download it to the PLC online. Afterwards, they check the results and supervise the process they are controlling through a Webcam. The experimental stations are illuminated by a lamp which is switched from a digital output of the PLC. The available experimental stations are: (1) a conveyor belt driven by a stepper motor; (2) a DC motor driven by a DC/DC Converter; (3) a three-phase induction motor with a variable speed drive; and (4) a temperature regulator with a Resistance Temperature Detector (RTD) TR Pt-100. The stations communicate with the PLC either directly or through converters and sensors. The aim of these experiments is to design different control strategies using PLC and to be familiar with real industrial applications.

Another example is found in [35, 36], where a remote laboratory was developed for industrial real systems among them: a quadruple-tank industrial scale model to study SISO and MIMO systems (Figure 11); an electro-pneumatic system with an IRB 1400 S4 robot from ABB; an industrial pilot plant with two reactors and three associated utilities circuits (heat, steam, and freeze); and an AC motor with a variable frequency drive. The purpose is to teach students automatic control principles as well as programming PLCs. Thus, students work with two environments simultaneously. First, they program the PLC by writing the control algorithm in a software environment provided by the manufacturer, which students must install on their computer previously and link it to the address and port of the selected PLC. Then, they debug the control strategy online and monitor the status of the process by means of a SCADA system—a conjunction of motorized cameras and a GUI. Four remote PLCs from different manufacturers are available and compatible with all experimental stations: Modicon PremiumTSX P57 254M PLC from Schneider Electric, SIMATIC S7-300 PLC from Siemens, SNAP-LCM4 PLC from Opto 22, and PLC XC-CPU-201 PLC from Moeller. PLCs communicates with the controlling server using OPC standard and with the instruments using Profibus protocol. Students have access to theoretical and practical documentation and instrumentation data sheet.



http://wwwlra.unileon.es/ControlLRA/ControlMaquetas/subpracticas/sub2.aspx - Microsoft Internet Explorer

Pump Conv. 0
Valve Pos. 0
Setpoint 50
Kp 10
I 0
D 0
Disturbance 0

Figure 11. Remote laboratory for industrial quadruple-tank [35].

2.11. **Embedded Systems**

Unlike general-purpose computers or Personal Computers (PCs), embedded systems are computers with a specific task within a larger system, which are extremely reliable, often mass-produced, benefiting from economies of scale, and may lack human interaction. Typically, their core unit is a microprocessor, which is a multipurpose programmable device that includes an Arithmetic Logic Unit (ALU) for mathematical operations and a control unit for retrieving instructions stored in its memory and processing accordingly the input digital signals through the ALU. The Central Processing Unit (CPU) of an embedded system could be a single microprocessor or more in case of multi-core processors. DSP is an optimized type of microprocessors with higher clock speeds, mathematical operations capacity, and also power consumption.

Other features of an embedded system are: volatile memory such as Random-Access Memory (RAM); non-volatile memory such as Read-Only Memory (ROM) and flash memory; I/P and O/P peripherals such as switches, relays, solenoids, LEDs, Liquid-Crystal Displays (LCDs), RF devices, sensors, D/A converters, A/D converters, timers, motors, and PWMs; support for communications standards such a Synchronous Serial Communication (SSC), USB, networks, Fieldbuses, Serial Communication Interfaces (SCIs); and boundary scan support using Joint Test Action Group (JTAG) or IEEE 1149.1. Having all these features on a single board or an IC is referred to as a Microcontroller (μC or MCU). Compilers and assemblers are used to convert high-level language and assembly language codes or respectively into machine language. Firmware such as the ROM Basic I/O System (BIOS) may either: contain only elementary basic functions of a device; provide services to higher-level software; or be the only program that will run on the system and provide all of its functions. Higher level Operating Systems (OSs) are also adopted such as Microsoft Disk Operating System (MSDOS), Linux, NET Berkeley Software Distribution BSD (NetBSD), and Open Services Gateway initiative (OSGi). As well, embedded Real-Time OSs (RTOSs) are adopted such as μC/OS-II, QNX, Windows CE, VxWorks, LynxOS, Operating System Embedded (OSE), and RTLinux.

A System on Ship (SoC) is complete system embedded on a single chip substrate that might consist of multiple processors, memory blocks, oscillators, voltage regulators, multipliers, interfaces and other peripherals. SOCs can be implemented as an ASIC using a Field-Programmable Gate Array (FPGA). FPGA is similar to CPLD but meant for more capacity, complex designs, design flexibility, and complex embedded functions and blocks. Microprocessor's block can be added within FPGAs to enable: applying a combination of serial and parallel processing to embedded system designs; reducing power consumption and dissipation; and forming a Programmable SoC (PSoC). Embedded microprocessors can be either hard-macro or soft core that are implemented using logic synthesis—such as Nios II from Altera (www.Altera.com) and MicroBlaze from Xilinx (www.Xilinx.com). FPGAs are programmed using HDLs (i.e., VHDL or Verilog).

In [37], a remote laboratory was developed for programming a AT89S51 microcontroller evaluation board from Atmel (www.atmel.com), which has In-System Programming (ISP) facilities, Figure 12. The board is equipped with I/O peripherals such as an LCD, stepper motors, keyboard, and LEDs. An embedded Web server is built using an AT89S52 microcontroller and serves as a user interface (UI) to control the experiment microcontroller board via internet. Users upload binary files to the target board, observe the changes via a Webcam and interact with the target board through virtual keyboard. A chat application is available within the software of the embedded Web server. The embedded Web server receives the code from users in form of a binary file and upload it to the flash memory of the experiment microcontroller board. The embedded Web server, additionally, sends log report to the database server and sends messages to the chat application.

2.12. Systems, Mechatronics, & Robotics

Systems engineering integrates several disciplines of engineering and management to enable the planning, design development, implementation, and maintenance of complex engineering projects over their life cycles. Work processes, optimization methods, risk management tools, and all other likely business and technical aspects of a project or system are considered and integrated in system engineering.

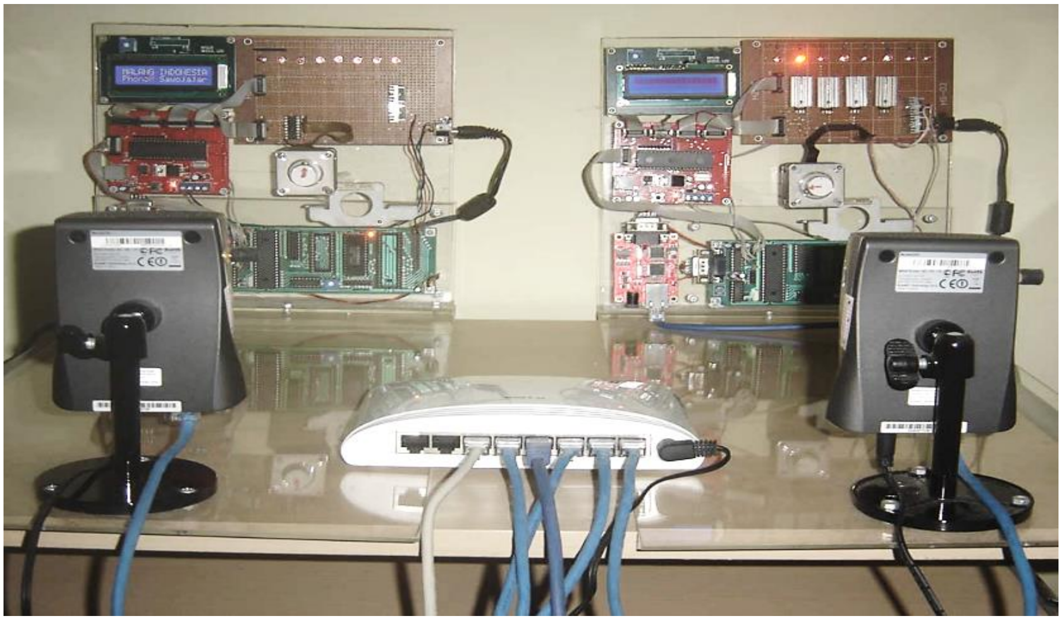
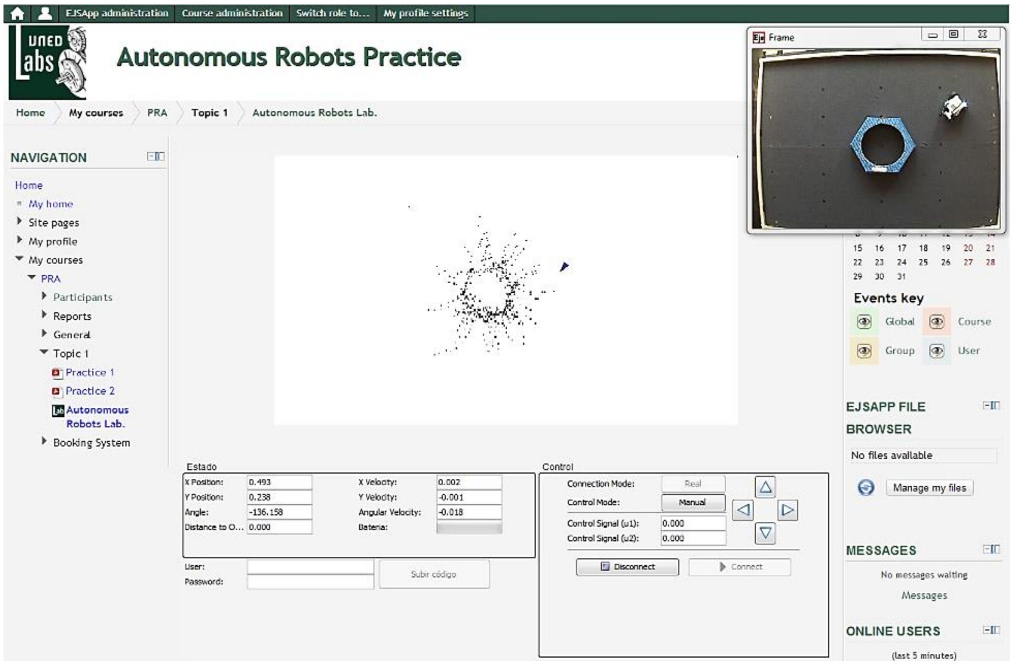


Figure 12. Remote laboratory for programming a microcontroller [37].

Mechatronics engineering, like systems engineering, is a multidisciplinary field of engineering but it focuses on smaller details rather than larger generalizations and relationships. Mechatronic design deals with the integrated and optimal design of a mechanical system controlled with embedded electronic components. Thus, a mechatronic system consists of a mechanical part that performs certain motions and an electronic part that controls and adds intelligence to the system using control and artificial intelligence techniques.

Mechatronics and robotics are distinguished by their dexterous manipulation capability in that robots can work, position, and move tools and other objects with far greater dexterity than other electromechanical machine. Mass production of products in heavy industries such as aerospace and automotive are no longer possible without using robots while meeting currently accepted quality and cost levels. Robots preform jobs more cheaply, accurately and reliably than humans with high endurance, speed, and precision. They can take the place of humans in dangerous environments or manufacturing processes, or resemble humans in appearance, behavior, and/or cognition. Robots could be android or humanoid robots that look like humans or stationary robots including robot arms that operate on the factory floor performing a wide variety of tasks.

In [38], a remote laboratory was developed, which allows users to tele-operate a LEGO Mindstorms NXT (http://mindstorms.lego.com) robot and take measurements from its sensors in order to compute the robot position and to obtain a map of its environment with an Occupancy Grid Mapping method, Figure 13. The ultrasonic sensor of the robot enables it to detect obstacles in a range from 0 to 255 cm and the two servo motors connected to the wheels allow the robot to move and/or rotate in the floor surface. The robot communicates with the laboratory server using Bluetooth. The laboratory is equipped with a high resolution camera that computes the position of the robot over the floor and a Webcam that gives students the visual feedback of the robot operation. The robot is equipped with three high bright LEDs that form an isosceles triangle, which is used to compute its position. First, a video frame is captured then the image is processed and the centroid of each individual LED is computed. Once the corrected positions of the points are computed, the image is transformed from camera frame to floor coordinates, which finally yields the real positions of the lights on the surface where the robot moves. Since the positions of the LEDs are known, it is straight forward to compute the center of mass of the robot and its orientation from the positions of the triangle vertex, and then estimate the velocities by a finite difference. It is possible to take manual control of the robot, which allows to send the movement commands directly to it by using the arrow buttons, or to execute it in the automatic mode, where the behavior of the robot is determined by the signal processing and control algorithms previously written and uploaded by students. One possible configuration of the system is built by following the ready-to-use structure, described in the NXT user guide (8547), with some minor modifications like adding three lights in order to be tracked by the artificial vision subsystem. A simple and intuitive GUI is provided, which contains the state of the system; the positions, the velocities, the state of the battery, and the error. By using other types of sensors equipped in the robot, other configurations are possible, which yields interesting practices such as multi-sensor fusion, Inertial Navigation System (INS), Kalman filter, and visual odometry.



X Position: 0.493
Y Position: 0.238
Angle: -136.158
Distance to O... 0.000
X Velocity: 0.002
Y Velocity: -0.001
Angular Velocity: -0.018
Bateria:
User:
Password:
Connection Mode: Real
Control Mode: Manual
Control Signal (u1): 0.000
Control Signal (u2): 0.000

Figure 13. Remote laboratory of a LEGO Mindstorms NXT robot [38].

2.13. **Biomedical Engineering**

Biomedical engineering deals with the application of engineering in biological systems or living organisms (e.g., human, animal, and plant) in order to achieve medical innovations and advance healthcare treatment (i.e., including diagnosis, monitoring, and therapy), and other areas that improve the living standards of societies, as well as to design and manufacture products that can monitor physiologic functions and assist in the diagnosis and treatment of patients. Engineering and technology principles are applied to understand, mimic, modify and control biological systems.

Disciplines within biomedical engineering could be classified into fundamentals and applications. Fundamentals encompass biomedical system analysis, biomechanics, biomaterials, and bioelectronics, whilst applications encompass medical devices, diagnostic equipment design, surgery, rehabilitation engineering and prosthetics design, and clinical engineering. Prominent biomedical engineering applications include the development of various diagnostic and therapeutic medical devices ranging from clinical equipment to micro-implants, common imaging equipment such as Magnetic Resonance Imaging (MRI) and Electroencephalography (EEG), biocompatible prostheses, regenerative tissue growth, and pharmaceutical drugs.

Remote laboratories applications in this field are very scarce yet. However, simple remote application approaches have been made, which are likely to be a bread ground for further developments. For example, in [39], a heart monitoring system was developed for diagnosing different kinds of cardiovascular diseases. It is capable of detecting and distinguishing the normal heart signal from an abnormal one. The sound signal of the heart beat is acquired using a good quality stethoscope with a micro-phone connected at its end to convert sound signal into electrical signal, in terms of capacitance change, for processing. The electrical signal is then converted into voltage signal and afterwards filtered, processed, and conditioned to get a Phonocardiogram (PCG). PCG is a plot of high fidelity recording of the sounds and murmurs made by the heart with the help of audio recording of the heart during a cardiac cycle. In contrast, the ordinary stethoscope cannot detect such sounds or murmurs, and provides no record of their occurrence.

2.14. Multidisciplinary Commercial Solutions

Computer-controlled commercial kits and platforms for laboratory experiments are increasingly gaining a wide popularity particularly among remote laboratory developers. As seen in the previous examples, commercial products were adopted from companies like Altera, Xilinx, DefSim, and LEGO Mindstorms NXT. Of Particular interest is NI that is currently predominating the remote laboratories market share with its versatile educational and multidisciplinary platforms and software. All NI educational platforms are computer-controlled and powered by LabVIEW, which facilitates their adoption in remote laboratories applications with minor efforts. Moreover, they are integrated and often used simultaneously NI Multisim, which enables comparing theoretical calculations with real behavior. The common educational NI platforms are the following.

2.14.1. NI Educational Laboratory Virtual Instrumentation Suite (ELVIS) II

NI ELVIS II (www.ni.com/ni-elvis/) is a multidisciplinary engineering educational platform released by NI, Figure 14(a). It is integrated with 12 of the most commonly used laboratory instruments (i.e., Oscilloscope, multimeter (DMM), function generator, power supply, Dynamic Signal Analyzer (DSA), bode analyzer, 2- and 3-wire current-voltage analyzer, Arbitrary Waveform Generator (AWG), digital reader/writer, and impedance analyzer) in a single compact form designed for education.

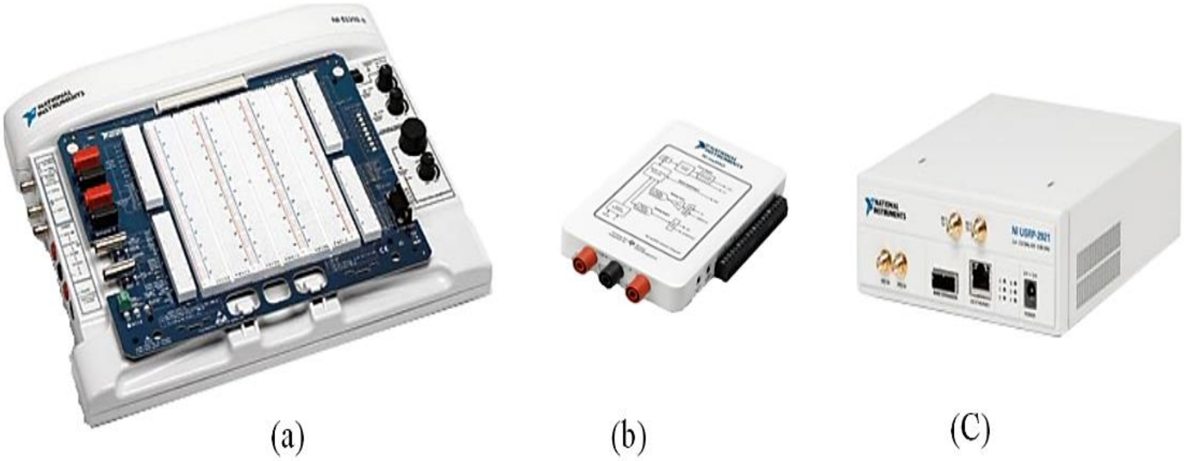


Figure 14. Commercial Multidisciplinary experimentation platforms from NI.

It has a variety of experiment plug-in boards and kits from NI and from other third-party companies such as:

- **NI digital electronics FPGA board:** for teaching digital electronics and FPGA using an XA Spartan-3E FPGA from Xilinx which is fully programmable with either LabVIEW software or Xilinx Integrated Software Environment (ISE) tools. The board includes the necessary peripherals such as LEDs, Dual in-Line Package (DIP) switches, push buttons, SSDs, and encoders.
- **Emona (www.emona.com) boards:**
  - *FOTEx:* for teaching fiber-optic telecommunications topics such as PCM, sampling, Time Division Multiple Access (TDMA), line coding and bit-clock regeneration, fiber optic

transmission, optical signal filtering, splitting and combining, fiber optic-bidirectional communication, and Wavelength-Division Multiple Access (WDMA). It includes over 20 circuit blocks that are used by students to hard wire a communications system with plug-in fiber-optic patch cables and examine the properties of fiber-optic communication.

- *DATEx:* for teaching digital and analog communications topics such as AM, FM, Double-Sideband (DSB) Modulation, Single-Sideband (SSB) Modulation, Pulse-Amplitude Modulation (PAM), Pulse-Code Modulation (PCM), PWM, speech, Signal-to-Noise Ratio (SNR), equation modeling, sampling, TDM, ASK, FSK, QAM, DSSS, noise. It also includes over 20 circuit blocks.
- *HELEx:* for teaching green engineering topics such as solar cell characteristics, configuration and performance, electrolysis, hydrogen fuel cell use, and power plant modeling.
- *SIGEx:* for teaching signals processing and systems topics such as characterizing linear and nonlinear signals, understanding convolution, using poles and zeros in the Laplace domain, and sampling and aliasing.

- **Quanser (www.quanser.com) QNET boards:**
  - *VTOL:* for teaching motion control, aerospace dynamics, and kinematics using a first Degree of Freedom (1 DOF) helicopter.
  - *DC Motor:* for teaching fundamentals of DC motor control by configuring motor position and speed, parameter estimation, and control of a haptic knob.
  - *Mechatronic Sensors:* for teaching physical properties of the 10 most commonly used analog and digital sensors. It features four digital sensors: Single-Pole Double-Throw (SPDT) microswitch, transmissive optical switch, reflective optical switch, and magnetic Hall Effect switch. It also features six analog sensors: potentiometer, optical analog distance, magnetic analog field, pressure, temperature, and piezo film.
  - *HVAC:* for teaching fluid dynamics and thermodynamics control by designing a control system that regulates temperature in a chamber equipped with an integrated electronic temperature sensor, 12 V halogen lamp with variable intensity, and a fan.

   - *Myoelectric:* for teaching principles of Electromyography (EMG) and basic signal processing functions and control schemes in biomedical engineering including filtering, rectification, integration, deadzone, and hysteresis. It features a PWM servo motor with built-in power amplifier and “gripper” for simulating a prosthetic actuator.
   - *Rotary Inverted Pendulum:* for teaching control topics such as system modeling, parameter estimation, balance control, Linear-Quadratic Regulator (LQR) design, nonlinear swing-up control, and energy-based design.
- **Freescale (www.freescale.com) microcontroller prototype board:** for teaching microcontroller theory using a Freescale HCS12 Microcontroller.

Unlimited number of applications and experiments can be mounted on the plug-in boards. Examples are found in [18, 20, 23, 28, 30].However, the instruments can be controlled by hands, and the plug-in boards can be used separately without the NI ELVIS II platform.

2.14.2. NI myDAQ

NI myDAQ (www.ni.com/mydaq/) is compact, portable, pocket-size, and low-cost Data Acquisition (DAQ) device powered by LabVIEW and combines 8 ready-to-run software-defined instruments—including power supply, function generator, AWG, oscilloscope, bode analyzer, DSA, digital reader/writer, and DMM—in a compact USB device, Figure 14(b). It includes two analog inputs (200 kS/s, 16 bits, ±10 V), which can be configured as either stereo line level inputs in audio mode or input for the oscilloscope, the DSA, or the bode analyzer instrument. It also includes two analog outputs (200 kS/s, 16 bits, ±10 V), which can be configured as either audio output or output for the function generator, the AWG, or the bode analyzer instrument. There are 8 digital I/O Programmable Function Interface (PFI) lines, which can be configured as a general-purpose software-timed digital I/P, or it can act as a special function I/P for a digital counter (3.3 V TTL-compatible). There are three power supplies (up to 500 mW of power) available for use. +15 V and –15 V can be used to power analog components while +5 V can be used to power digital components. NI miniSystems extend hardware capabilities of NI myDAQ with application-specific add-ons and accessories from NI and from other third-party companies such as:

- **Pitsco Education (www.pitsco.com):**
  - *myQuake*: accelerometer measurement and motion control system for applications such as resonant frequencies, sensor measurement, and data acquisition and analysis.
  - *myVTO*: a 1 DOF helicopter plant with a PC fan to hover and a Hall effect distance measuring sensor for teaching concepts of flight dynamics, motion control, and PID control.
  - *myTemp*: a multisensor temperature measurement and control system using Negative Temperature Coefficient (NTC) thermistors for teaching temperature process control and heat transfer, conduction, convection, and radiation.
- **Elenco (www.elenco.com**):
  - myGrid: an AC power grid which includes a traditional power source through a motor representing fossil fuels and a renewable power source using a solar panel. It models how power is generated, transmitted, and distributed in a residential setting across three houses and allows monitoring it.
  - *Protoboard*: breadboard to connect directly to myDAQ hardware.
- **Sparkfun (www.sparkfun.com):**
  - *Protoboard*: breadboard to connect directly to myDAQ hardware.
- **iWorx (www.iworx.com):**
  - *IX-myDAQ Sensor Adapter*: an adapter board for interfacing a wide range of iWorx sensors and transducers.
- **Florida Research Instruments (www.floridaresearchinstruments.com):**
  - *BNC Probe Adapter*: provides industry-standard Bayonet Neill–Concelman (BNC) connectors for quick connection to myDAQ analog I/Os.

2.14.3. NI Universal Software Radio Peripheral (USRP)

NI USRP-292x (www.ni.com/usrp/) is an affordable, flexible radio connected to PC for prototyping wireless RF and communications systems, Figure 14(c). Necessary signal processing and programming building blocks in LabVIEW are provided for designing both transmit and receive chain for physical layer and basic Media Access Control (MAC) layer implementations. It has a tunable center frequency from 50 MHz to 2.2 GHz, 400MHz- 4.4GHz, 2.4GHz-2.5GHz, and 4.9GHz-5.9GHz covering FM radio, Global Positioning System (GPS), Global System for Mobile Communications (GSM), radar, and Industrial Scientific and Medical (ISM) bands. It acts as an ideal spectrum monitor with up to 40 MHz bandwidth, as well as an affordable record and playback solution with up to 20 MHz bandwidth. Combining two NI USRP devices allows building a 2x2 MIMO prototyping system.

Other relevant products from NI, not specifically for educational purpose, that have been widely adopted in remote laboratories applications are NI CompactRIO (www.ni.com/compactrio/), NI CompactDAQ (www.ni.com/data-acquisition/compactdaq/), and PCI eXtensions for Instrumentation (PXI) (www.ni.com/pxi/).

2.15. Conclusion

In this chapter a profound review on remote laboratories applications in electrical engineering education was provided. The chapter addressed the milestones and the most remarkable achievements in each sub-discipline of electrical engineering. The goal is to identify the weaknesses and strengths regarding to each application type (e.g., high power or synchronous real-time applications have different requirement than low power or asynchronous applications, respectively) and to what extent remote laboratories could replace their traditional counterpart in each application type. On the other hand, this study allowed understanding the technical architecture of remote laboratories and defining their common components independently from the kind of application. This was very necessary for defining the LaaS paradigm and the modular architecture, which is discussed in Chapter 5.

Despite their few years of age, remote laboratories have rapidly proliferated in all electrical engineering sub-disciplines as well as in other multidisciplinary fields such as biomedical engineering and system engineering. Application range varied from very small or micro-scale applications such as microelectronics field to high-voltage applications such as power systems and electrical machines. Issues concerning online control, switching, monitoring, scheduling, and administration have been overcome in many solutions.

Generally speaking, there were no notable reported limitation in their implementations and everything student can do in a traditional laboratory was possible to be done in remote laboratory arrangement. For instance, in some applications it was possible to remotely program devices, wire circuits, moving robot arms, etc. moreover, some applications managed creating a collaborative environment online using video conferences and chat tools. Besides, proponents of remote laboratories argue that nowadays industrial processes are commonly automated and controlled remotely. Adding to this the inherent features of remote laboratories such as: providing a safe experimentation environment, accessing with neither temporal nor geographical constraints, sharing laboratory resources among universities and institutions, and offsetting high cost of unaffordable equipment.

These factors altogether justify their rapid proliferation and the adoption as an effective solution and foresee the predominance of remote laboratories in the next few years and their replacement for traditional laboratories in the near future. Interestingly, the recent interest of corporations facilitated remote laboratories adoptions and yielded more effective commercial solutions and platforms such as NI ELVIS II, NI USRP, and NI myDAQ.

> *i.e., it is worth noting that other disciplines of science—such as life sciences and physical sciences—have been, so far, reluctant to adopt remote laboratories. The main possible reason is that most of the reported remote laboratories have been developed by research engineers and used/maintained by the same group for a specific course. The required engineering curriculum for developing such laboratories is not usually available among research groups from any field of life of physical sciences. The second possible reason is that in disciplines—such as pharmacology, chemistry, and biology—substances and elements might be consumed for each experiment session without possible reutilization. This will require a charging*

mechanism to allow users to pay for what they consume during the sessions, which is actually hasn’t been developed or reported yet in the literature.

This page intentionally left blank

Chapter 3

3. Development and Implementation of Remote Electronics Experiments

This chapter reports on the development and implementation of remote electronics experiments using Virtual Instrument Systems in Reality (VISIR) [40, 41]. VISIR is a remote laboratory for wiring and measuring electronics circuits on a breadboard remotely. The user designs and constructs his/her circuit via PC-mouse on a seamlessly simulated workbench that resembles the real lab elements and components. Once the designed circuit is submitted, it is first sent to be verified, then to be wired and measured by real instruments, and finally, it is received by the user on his/her PC-screen in real-time. The signal processing department at BTH in Sweden together with NI in the USA (as a supplier of equipment) and Axiom EduTECH (www.vtiinstruments.com) in Sweden (as a supplier of education, technical software, and engineering services for noise and vibration analysis) launched the VISIR project (http://openlabs.bth.se/electronics) at the end of 2006. The project was financially supported by BTH and the Swedish governmental agency for innovation systems.

So far, six universities have already implemented VISIR after BTH: Carinthia University of Applied Sciences and FH Campus Wien for Applied Sciences, both in Austria; ISEP in Portugal; University of Deusto and UNED, both in Spain and; and Madras Institute of Technology (IIT-M) in India. The following Universities have shown their interest in participating in this project but they have not yet implemented it: University of Genoa in Italy; Princess Sumaya University for Technology in Jordan; Gunadarma University in Indonesia; Institute for the Development of New

Technologies in Portugal; and College of the North Atlantica in Qatar. A Special Interest Group of VISIR (SIG VISIR) [42] was created by the International Association of Online Engineering (IAOE) (www.online-engineering.org) in order to foster the collaboration within the community and to support the project dissemination.

3.1. Hardware Description

3.1.1. PXI Platform

The instrumentation platform of VISIR is based on NI PXI platform, which consists of instrument module cards, a controller card, and a chassis into which all the cards are plugged. For every component, there are various models depending on its technical characteristics and there is a flexibility to choose among them. Table 1 illustrates the components of the PXI platform currently installed at UNED.

3.1.2. Relay Switching Matrix

The relay switching matrix is a stack of "PCI/104" (www.pc104.org) sized boards, manufactured at BTH, which controls the terminals connection of the components and the NI PXI-modules. It acts as a circuit-wiring robot and it is designed for low frequency analog electronic circuits' experiments. It consists of two types of boards: (1) instrument boards, which handle the connection of their corresponding NI PXI-module; and (2) component boards, which handle the connection of the components installed in them. The NI PXI-chassis is connected to the relay switching matrix by a USB cable and the terminals of the NI PXI-modules are connected to it by either coaxial cables or cords. Each component board comprises 10 sockets and each socket is connected to a DPST relay—four of these sockets can be connected instead to 2 Single-Pole Single-Throw (SPST) relays each. Thus, a matrix can contain up to 16×10 DPST relays. Two leads components occupy one socket, while more leads components occupy more sockets. In addition, two 20-pin IC sockets for complex circuit connections are provided in each component board. A component board is shown in Figure 15.

Table 1. PXI platform installed at UNED.

	NI PXI-Chassis	NI PXI-Modules	NI PXI-Controller
Description	It is the backbone of the PXI platform into which all the instrument cards (NI PXI-modules) and the NI PXI-controller are plugged.	The instrument cards that substitute the real instruments. They are plugged into the NI PXI-Chassis. All these cards can be added and removed depending on the demands.	It is an embedded PC plugged into the NI PXI-Chassis. It includes an integrated CPU, a hard drive, RAM, Ethernet, video card, keyboard/mouse connectors, serial, USB, Microsoft windows software etc. All these features are integrated. Hence, it eliminates the need for an external PC but, it could be replaced by a PC.
Model installed at UNED	NI PXI-Chassis (NI PXI-1031)	NI PXI-DC Power Supply (NI PXI-4110) NI PXI-Digital Multi-Meter (NI PXI -4072) NI PXI-Function Generator (NI PXI-5412) NI PXI-Oscilloscope (NI PXI-5114)	NI-PXI Controller (NI PXI-8105)

According to the data sheet, the maximum carry current of the relays is 2 A and the minimum life expectancy is 3×10^8 operations (approximately two operations per second continuously for five years). Putting the relay switching matrix into a closed case is not recommended because it should be easy to swap components and rewire branches. However, it is very important to protect the relay switching matrix from non-qualified persons. The matrix contains a main Peripheral Interface Controller (PIC18F4550), in addition to a separate controller (PIC16F767) for each board, which sends commands to the relays of that board to open or close with regard to the received data The matrix creates circuits by manipulating—by opening and closing relays with regard to the received circuit design from the controller—the connection of the NI PXI-modules' terminals and the components' leads on a common 10 nodes (A-I, 0) propagating through all its, creating a node bus which acts as a breadboard as shown in Figure 16.

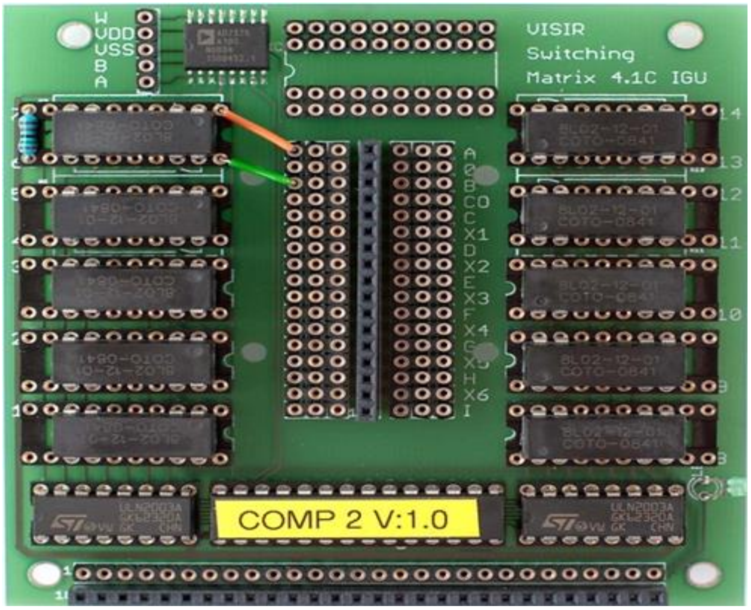


Figure 15. A component board with 10 DPST.

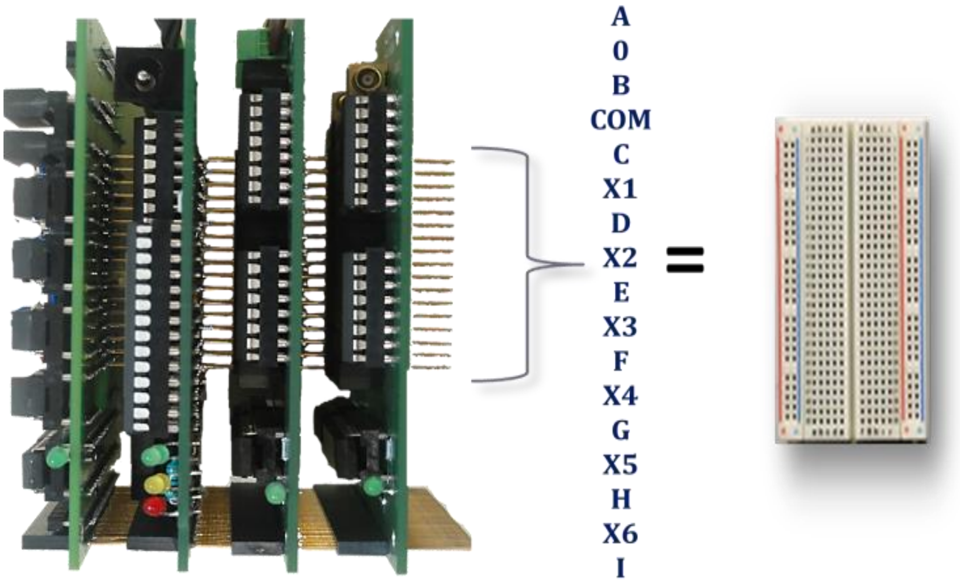


Figure 16. The common propagating nodes within the relay switching matrix.

The common nodes are divided into two group. The first contains the nodes from A to I, and 0 (GND). The second contains the nodes from X1 to X6, and "COM". The ground terminals of the function generator and the oscilloscope are hardwired to the node 0 (GND). The function generator output is hardwired to the node A. The oscilloscope channels, as well as the DMM channels, are dynamically connected to any node, depending on the user's circuit design. The Power supply connectors (0, COM, +6V, +20V, -20V, and AUX) are connected internally to the node 0 (GND) and to the nodes of the second group (COM, X1, X2, X3, X4), respectively, then they are connected to the first group either by a shortcut wire or by two relay switches, as the second group are not supported in the current software version. The complexity of the matrix depends on the number of nodes it has, e.g. from N nodes we can obtain N× [(N-1)/2] branches. The current relay switching matrix has 10 nodes (A-I, 0), which are appropriate for most electronic circuits' practices. The internal interconnection of the instrument boards is shown in Figure 17.

The component list file describes all the installed components and instruments in the relay switching matrix to make them known to the software. There is only one component list per relay switching matrix and each added component or instrument is listed in the file with regard to its value, the number of the relays on which it is mounted, the nodes to which it is connected, and the number of the board on which it is located. The user can only build circuits corresponding to the allowed connections in the component list file. The relays of the component boards are numbered as follows: 1, 2, 3, 5, 7, 8, 9, 10, 11, and 13 respectively, i.e. this is in the case that all the relays are DPST. In the case that the DPST relays (5, 7, 11, and 13) are replaced with SPST relays, the numbering will be from 1 to 14. Each board has an Inter-Integrated Circuit (I^2C) label, which corresponds to a number as shown in Table 2.

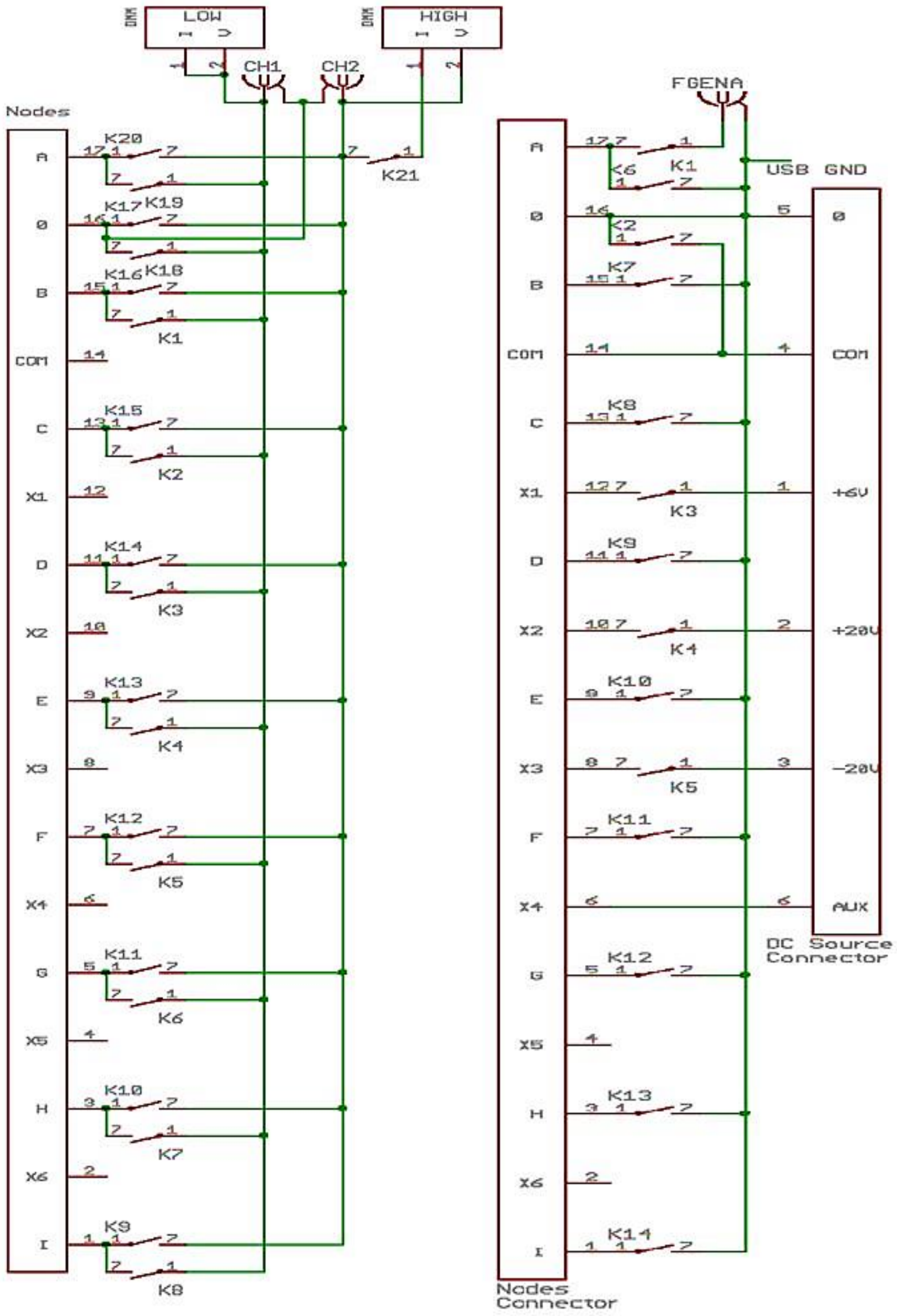


Figure 17. The internal interconnection of the instrument boards.

Table 2. Corresponding board number of each I^2C label.

Board Type	Board Number	I^2C Label
Component board 1	1	COMP 1
Component board 2	2	COMP 2
….etc.	…	…
Oscilloscope board	16	OSC 16
DMM board	17	DMM 17
Source board	24	SRC 24

For instance, "*R_2_7 AB 10K*" represents a resistor of 10k ohms installed on the relay number 7 of the board number 2 and connected to the nodes (A, B) as shown in Figure 2. The entire connection of VISIR is shown in Figure 18.

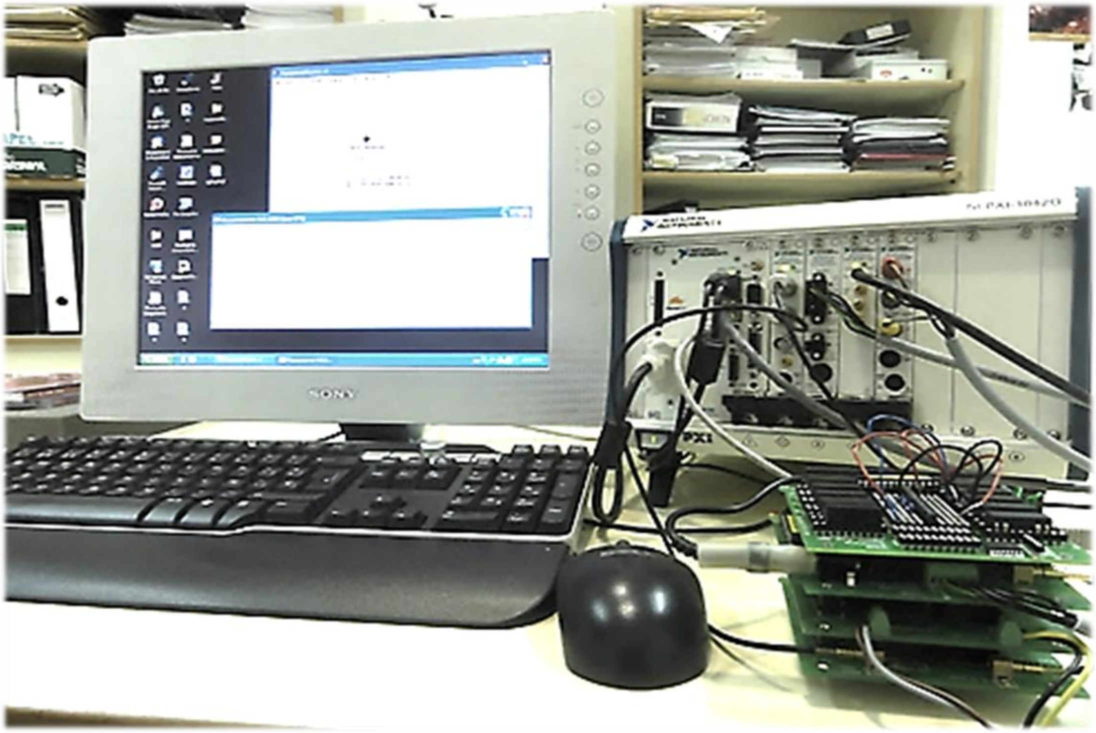


Figure 18. VISIR installed at UNED.

3.2. Software Description and Operation Cycle

The VISIR software is an open-source that is released under a GNU General Public License (GPL) and available at (http://svn.openlabs.bth.se/trac). In this section, a step-by-step presentation of the operation cycle is shown along with the description of the development of each software component.

3.2.1. User Interface (UI)

The UI is the frontal web page of VISIR that handles all the administration, access, and authentication process. It is written in the scripting language PHP in connection with a Relational Database Management System (RDMS) MySQL, and it is hosted in an Apache Hypertext Transfer Protocol (HTTP) Web server. It uses the secure protocol HTTP Secure (HTTPS) and provides many features similar to those provided by a Learning Management System (LMS) in order to facilitate the implementation of VISIR in the learning process. The capabilities and limitation of these features are associated with the account types. The properties and the privileges of each account type are explained as follows.

I. **Administrator account:** it is the account of the lab provider. VISIR can have one or more administrator account. The administrator account has the following privileges:
   - ➢ Create contents in the UI through the WIKI markup syntax.
   - ➢ Upload files, videos, manuals, etc.; this can be done by uploading them to the web server (Apache) and linking them to the PHP code of the Page.
   - ➢ Create, update, and delete courses associated with a start and an end date, a maximum number of users, and assigned teachers and/or instructor accounts.
   - ➢ Modify or remove any user account.
   - ➢ In any course, the administrator account can switch to "teacher view" to have all the teacher privileges in that course.

II. **Teacher account:** it is created by the administrator and associated with a certain course. It has the following privileges:

- Add and remove experiments; this is done by allowing certain components to appear to the student in each experiment.
- Add, remove, and modify student accounts.
- Make a teacher scheduled reservation; the teacher reserves a number of seats within an interval of time so that, she can put his/her students in groups and assign an instructor to each group. These seats are visible to the student with the teacher's name. Accordingly, the student chooses his/her group and reserves his/her seat.
- The teacher can switch to "student view" in order to view the contents as seen by his/her students.

III. **Student and instructor accounts:** they are created by the teacher and are associated with a certain course. They only allows access to the experiments that are created by the teacher within such course. The student and the instructor, likewise, can make a scheduled reservation separately or reserve a seat belonging to the scheduled reservation of the teacher.

IV. **Guest account:** it is a public limited trial account created by the administrator and it does not require registration. It can be utilized by anyone wanting to try an available experiment prepared by the teacher assigned to that account.

3.2.2. **Experiment Client**

The experiment client is a simulated workbench embedded in the Hypertext Markup Language (HTML) code of the user interface. It is written in ActionScript for Adobe Flash and located inside the same folder of the UI files, which are hosted by the Web server. The user chooses the instrument interface with which she is familiar regardless of the model or the manufacturer of the corresponding real instrument as shown in Figure 19.

In this way, it is possible to use a virtual front panel of an instrument model to control another instrument model as long as the performance of the real instrument is equal or better than the performance of the virtual instrument.

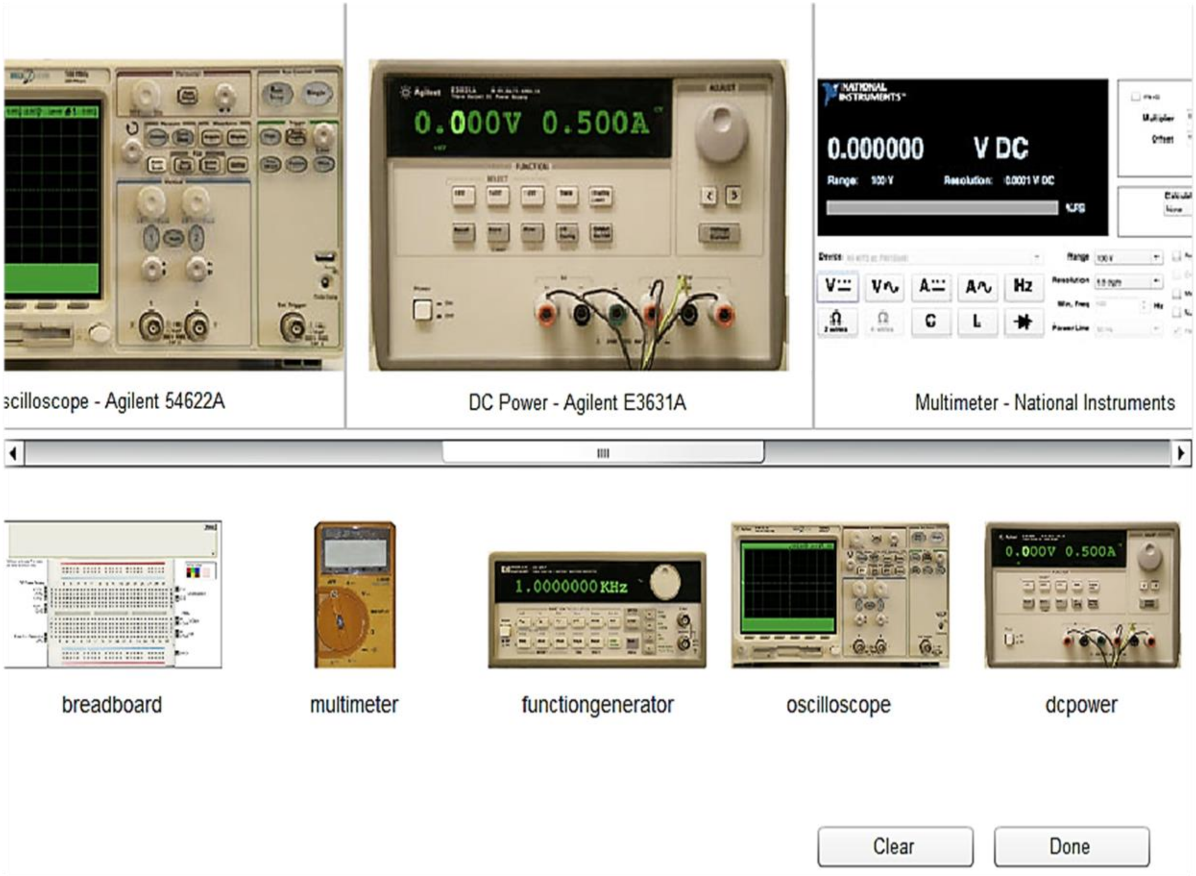


Figure 19. Selecting virtual interfaces of instrument from different models.

The available interfaces are:

- DMM (Fluke 23).
- A function generator (HP 33120A).
- An oscilloscope (Agilent 54622A).
- A DC Power Supply (E3631A).
- The default PXI-instruments interfaces from NI.
- Traditional breadboard.

After choosing the instrument interfaces, the user starts to design and wire his/her circuit on a simulated breadboard by clicking and dragging the simulated components and wires with his/her PC-mouse as shown in Figure 20.

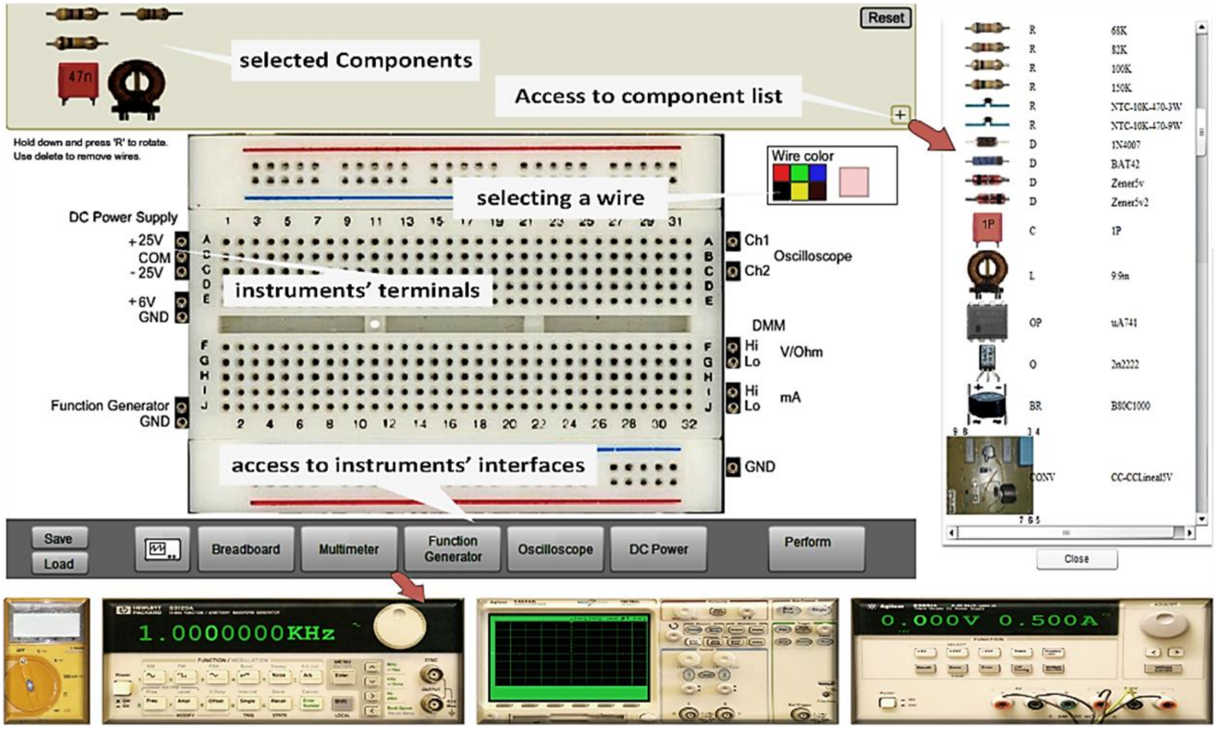
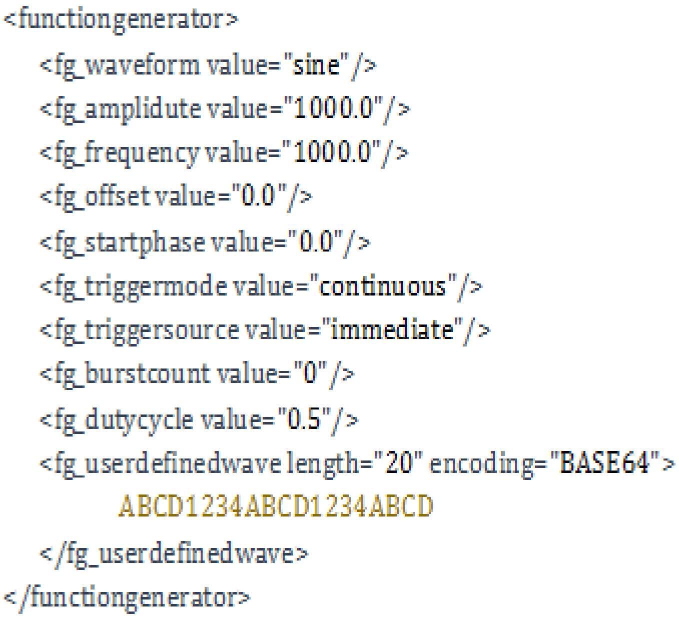


Figure 20. Simulated work bench of VISIR.

Once the user gets his/her circuit ready, with all the instrument values adjusted, and clicks on the "*perform experiment*" button. The experiment client sends the designed circuit with all the adjustments and the configurations to the "*measurement server*" (see the next subsection) through the experiment protocol. The experiment protocol is an Extensible Markup Language (XML)-based protocol, which uses either XML Socket or HTTP (the current configuration) over the TCP/IP model to transport the requested data to the "measurement server". For instance, an experiment protocol request sent by a function generator could look as shown in Figure 21.

The experiment protocol describes which settings and functions each instrument type can perform, independent of hardware manufacturer, thus it is possible to select an instrument simulated interface independently of the manufacturer, and also to create new interfaces of instruments that do not exist in the current set.



```
<functiongenerator>
   <fg_waveform value="sine"/>
   <fg_amplidute value="1000.0"/>
   <fg_frequency value="1000.0"/>
   <fg_offset value="0.0"/>
   <fg_startphase value="0.0"/>
   <fg_triggermode value="continuous"/>
   <fg_triggersource value="immediate"/>
   <fg_burstcount value="0"/>
   <fg_dutycycle value="0.5"/>
   <fg_userdefinedwave length="20" encoding="BASE64">
         ABCD1234ABCD1234ABCD
   </fg_userdefinedwave>
</functiongenerator>
```

Figure 21. A function generator message based on the experiment protocol.

3.2.3. **Measurement Server**

The measurement server is a software application written for Microsoft Windows in Microsoft Visual C++. Thus, the Microsoft runtime libraries (Microsoft Visual C++ Redistributable Package) should be installed before running it. It receives the measurement requests from the experiment client in separate TCP sessions and through the port 2324. Thus, connect and disconnect are required for every request made to the server. The requests/responses should not exceed 64 KB in size.

The role of the measurement server can be defined in the following steps:

- **Authentication:** at each request, it verifies that the client is a valid user by validating the client cookie generated by the Web server with the database.
- **Verification:** it acts as a virtual instructor and compares the received circuit data with the "*max lists*" before sending it to be executed on the real instruments, to avoid hazardous circuits or any damage that may be caused to the real instruments. A "*max list*" is a file

created by the teacher to define the permitted values of the components and the instruments in a certain experiment. This enables the teacher to be the sole person responsible for any damage. The "*max lists*" of all the available experiments are inserted inside the folder of the measurement server software. The max list of an experiment determines which connections are allowed in an experiment.

- **Queuing:** it can handle requests from 16 simultaneous clients in less than a second (1/16 second is the maximum time for each request) by queuing all the simultaneous requests and performing them sequentially with regard to the priority, reservation, etc.
- **Proxy server:** after validating and queuing the requests, it starts to send the requests sequentially in separate TCP sessions over TCP/IP through the port 5001 to the "*equipment server*" (see the next subsection). The measurement server acts as a gateway and could serve more than one equipment server.

3.2.4. Equipment Server

The equipment server consists of the PXI platform and the relay switching matrix. The equipment server software is installed in the NI PXI-controller (or in a separate PC instead). It is a software application for instrumentation control developed by LabVIEW. The equipment server software receives validated sequential experiment protocol requests from the measurement server in separate TCP sessions over TCP/IP through the port 5001 and it executes it through the connected instruments. After that, the results return back to the client PC-screen with the same sequence. The results are represented in the form of measurements on the virtual instrument interfaces.

Most of the undergraduate electronics laboratories of all the universities around the world have common equipment (oscilloscopes, function generators, DMMs, DC power supplies, and breadboards) regardless of their model and manufacturer. The current VISIR supports PXI, however, other universities would like to use another platform such as LAN eXtensions for Instrumentation (LXI) [43] or General Purpose Interface Bus (GPIB). To enable interchangeability between workbenches and different grid nodes (i.e., different universities), VISIR recommends the functions and attributes defined by the Interchangeable Virtual Instruments (IVI) foundation (www.ivifoundation.org) to be used to describe the base class capabilities and the class extension capabilities of the lab hardware. In fact, all the instrument drivers installed in the equipment server are IVI compliant. Accordingly, it is possible to create a standardized approach, as described in Figure 22, which is easy to adopt.

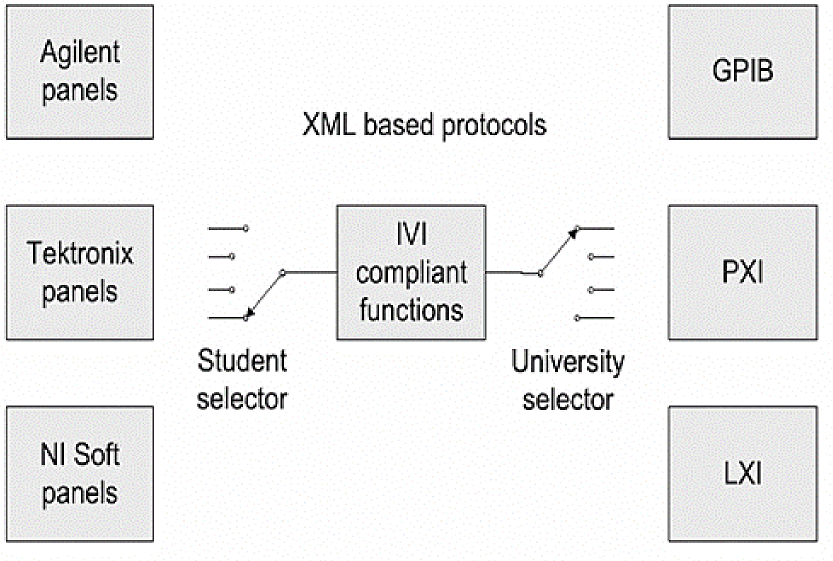


Figure 22. The role of IVI standard.

The base capabilities of an IVI-complaint instrument are the functions of class that are common to most of the instruments available in the class. For instance, for an oscilloscope the base capabilities mean edge triggering only but other triggering methods are defined as extension capabilities. The functions supported by the VISIR oscilloscope are listed in Table 3.

The goal of the IVI Foundation is to support 95 % of the instruments in a particular class. The Virtual Instrument Software Architecture (VISA) standard is accepted too, but the instrument functions should be those defined by the IVI standard. The component list file of the relay matrix is inserted inside the equipment server software folder to define the available components to the software. The whole operation cycle can be summarized as shown in Figure 23.

Table 3. IVI base and extension capabilities of an oscilloscope.

Group Name	Description
IviScopeBase	Base Capabilities of the IviScope specification. This group includes the capability to acquire waveforms using edge triggering.
IviScopeWaveformMeas	Extension of IviScope with the ability to calculate waveform measurements, such as rise time or frequency.
IviScopeTrigger Modifier	Extension of IviScope with the ability to modify the behavior of the triggering subsystem in the absence of an expected trigger.
IviScopeAuto- Setup	Extension of IviScope with the automatic configuration ability.

3.3. **Deployment and Results**[1]

VISIR was successfully implemented and deployed in undergraduate engineering practices at ISEP, University of Deusto, and UNED [44]. The overall results were satisfactory. Several electronic circuits' practices have been carried out by VISIR without any inconvenient. These practices include: half-wave rectifier with and without a filter; inverter and non-inverter operational amplifier; regulator with zener diode; and common emitter and common collector BJT. At ISEP, VISIR was deployed in a course of more than 270 students enrolled without many irregularities. The students stated that they did not feel that it helped with their motivation but they would like to extend its utilization to other subjects. They also felt that a formal tutorial could have been helpful. At University of Deusto, VISIR was accepted among the students as a useful tool for practical sessions; the students who had used it gained more self-confidence when they started using the real lab, even though, their first time was through a remote lab. On the other hand, the students considered it as a support tool, not a total substitution for real labs.

[1] In this section, I would like to acknowledge particularly the contribution of Prof. Gabriel Diaz from UNED.

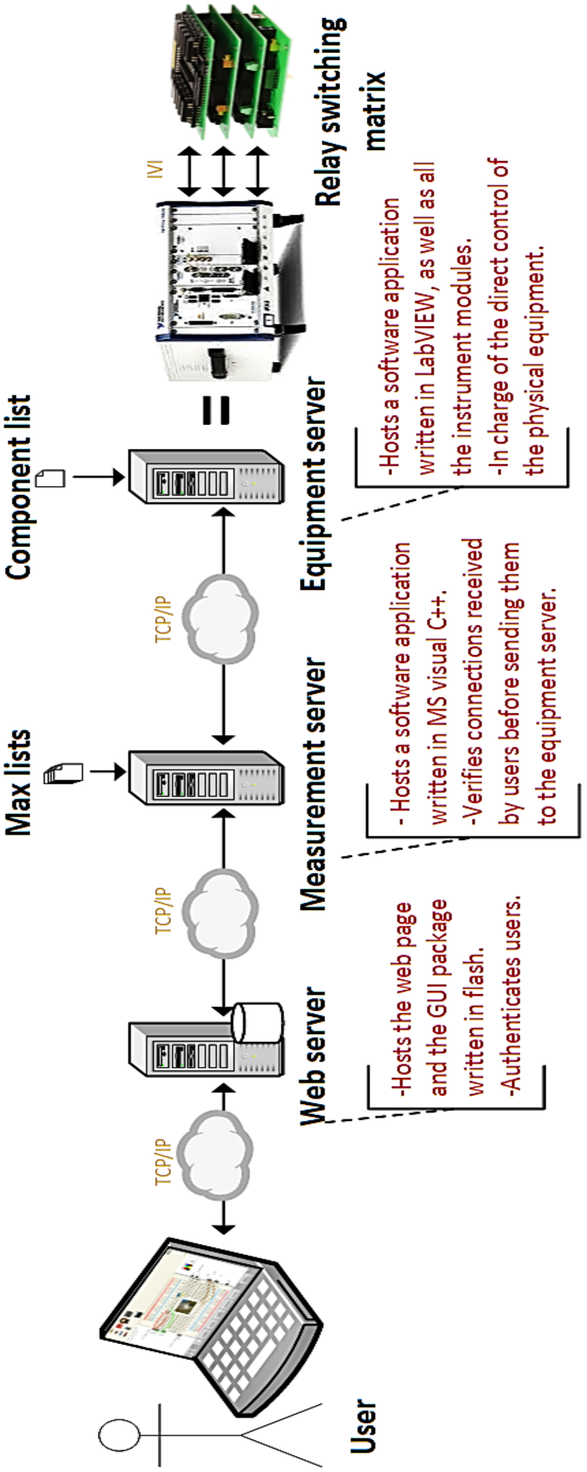

Component list
Max lists
TCP/IP
TCP/IP
TCP/IP
IVI
Relay switching matrix
Equipment server
Measurement server
Web server
User
-Hosts a software application written in LabVIEW, as well as all the instrument modules.
-In charge of the direct control of the physical equipment.
- Hosts a software application written in MS visual C++.
-Verifies connections received by users before sending them to the equipment server.
-Hosts the web page and the GUI package written in flash.
-Authenticates users.

Figure 23. VISIR components and overall operation cycle.

At UNED, during the academic year 2009/2010, the students of the subject "*Electronic Circuits and Components*", a first year subject of the Technical Industrial Engineering career in Spain, used the VISIR installed at University of Deusto, thanks to an agreement between both universities. The goal was to evaluate the system performance and to check its accuracy and sustainability when it comes to real time deployment. About 40 enrolled students were using the VISIR of Deusto during two entire days without any problems or inconvenience. The students could repeat their experiments varying the values all the time they needed during the two days. In other words, the results were very satisfactory and the system was proven to be sustainable. Many of these students stated that they would like to use VISIR for electronic circuit practices of other subjects. Many students also stated that it would be better to use VISIR to make the first approach towards the instruments and, afterwards, repeat the same practices in the real laboratory. Owing to this positive perception, the Electrical and Computer Engineering Department at UNED decided to install its own VISIR for the undergraduate electronic circuit's practices. The department installed a VISIR in December of 2010. The system has been in operation since the academic year 2010/2011 and was used as a mandatory pre-laboratory work for students of the subject, "*Foundations on Electronic Engineering*"− a subject within the new Electronic Engineering grade of Bologna that was recently applied in Spain. The goal was to use VISIR to take the first approach to the instruments and the typical ways of work in a real electronics laboratory. Afterwards, the students would repeat the same practices in the real laboratory at the department. This procedure allowed the students to gain more efficient experimentation skills during their first electronics practices. At the end of the first academic year of application (2010/2011), a survey was conducted among all the enrolled students (64 students) in that subject. The survey encompassed questions in terms of learning outcomes, sense of reality, and performance. The survey results are shown in Table 4.

As seen in the table, the performance of the system is notably high. The sense of reality could be better – this may owe to the lack of live view (Web cam) of the equipment, since in VISIR student only see the simulated workbench even though she is manipulating a real one – but it is still relatively high for being a remote laboratory. In general terms, VISIR was definitely feasible and students enjoyed the experience and gained a higher motivation for learning. On the other hand, many feedback was gathered from administrators and teachers, from universities within the VISIR community, who have been working with VISIR. In addition to the common positive perception, some drawbacks and limits have been addressed such as: lack of assessment and evaluation, limited number of component boards in the relay switching matrix, and limited number of available instruments.

Table 4. Survey on VISIR Deployment at UNED.

	Question	**Score (out of 5)**	**Average (%)**
Feasibility	*Q.1.* It helped me to understand the subject contents?	3.76	**77.53**
	Q.2. It was useful for trying many circuits without any fear of errors?	4.82	
	Q.3. I recommend it to be used in other subjects?	3.05	
Sense of reality	*Q.4.* I felt that I was working with a real laboratory environment and not a simulation?	3.69	**73.8**
Performance	*Q.5.* It was responding without time delay?	4.49	**86**
	Q.6. There were no software bugs or access errors?	4.11	

3.4. Implementation in a MOOC at UNED[1]

Beyond the ordinary implementation of VISIR in undergraduate engineering practices, a new MOOC (i.e., or COMA in Spanish) has been provided by the Electrical and Computer Engineering Department at UNED, which adopts VISIR as a core component of all its modules. It is open to the public all around the world and tailored for apprentices and workers. It's current and first version—

[1] In this section, I would like to acknowledge particularly the contribution of Felix Loro-Garcia and Prof. Gabriel Diaz from UNED.

denominated “*Bases de Circuitos y Electrónica Práctica*”—is pilot and taught in Spanish [45]. Based on the compiled feedback, a new advanced and English version will be released. The pilot course was launched in May 2013 with more than 2200 enrolled students and is due to finish in September 2013. It is structured in 8 modules of 10 hours. The first module introduces circuit simulation with tools such as SPICE and Micro-Cap and the subsequent modules involves real-time practicing with VISIR, so that students can compare between simulation and real-time results. A set of tutorial videos—as seen in Figure 24—are provided in each module followed by questions and practical tasks.

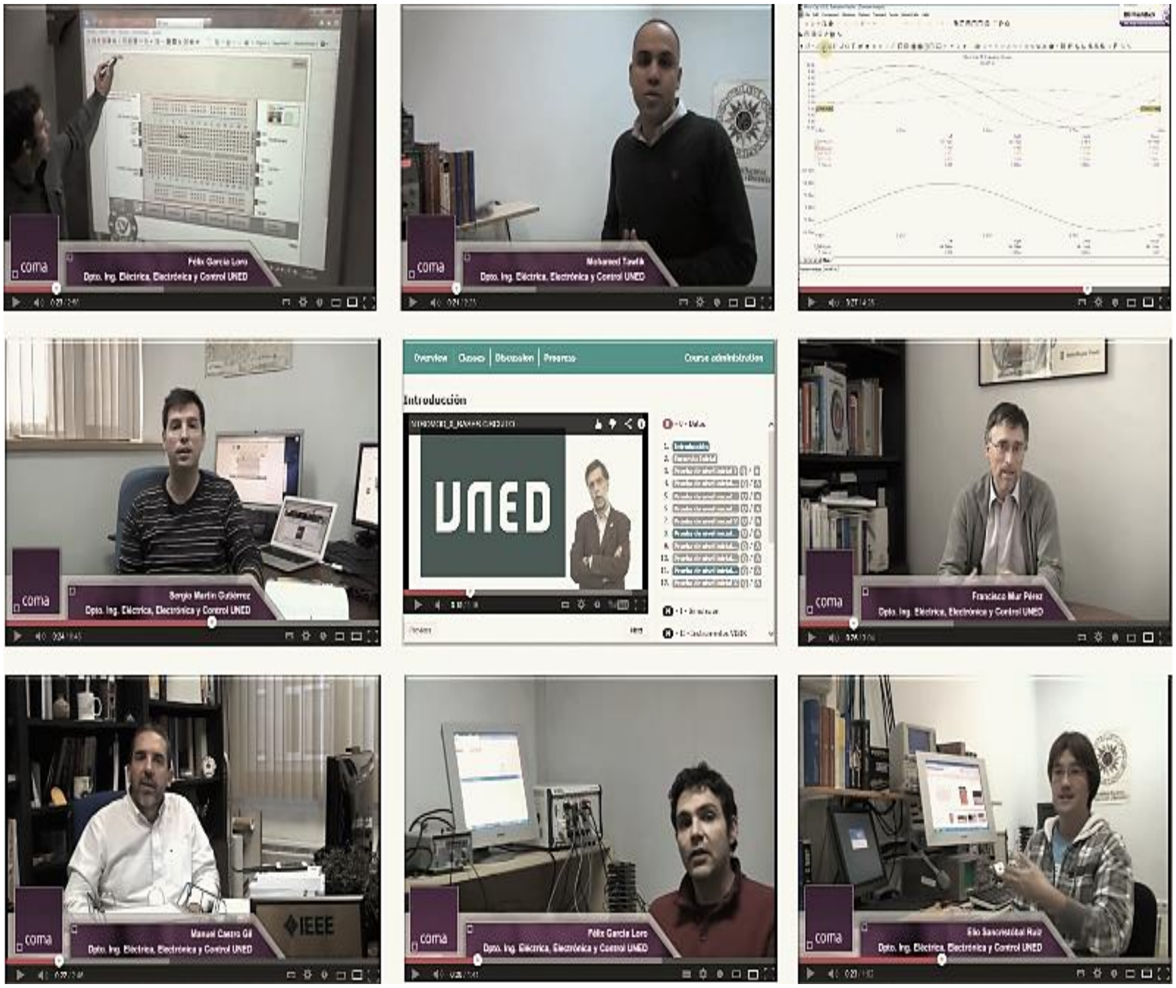

coma
Félix García Loro
Dpto. Ing. Eléctrica, Electrónica y Control UNED
coma
Mohamed Tawfik
Dpto. Ing. Eléctrica, Electrónica y Control UNED
coma
Sergio Martín Gutiérrez
Dpto. Ing. Eléctrica, Electrónica y Control UNED
Overview
Classes
Discussion
Progress
Course administration
Introducción
UNED
coma
Francisco Mur Pérez
Dpto. Ing. Eléctrica, Electrónica y Control UNED
coma
Manuel Castro Gil
Dpto. Ing. Eléctrica, Electrónica y Control UNED
IEEE
coma
Félix García Loro
Dpto. Ing. Eléctrica, Electrónica y Control UNED
coma
Elio Sancristobal Ruiz
Dpto. Ing. Eléctrica, Electrónica y Control UNED

Figure 24. Video tutorials in UNED-COMA.

The practices are almost the same provided in the undergraduate electronics course. The theoretical MOOC course "*Circuits & Electronics 6.002x*" [46] provided by MITx is recommended as the only perquisite for this MOOC. A preliminary survey about the profile of the enrolled students was taken among more than 1100 participants before the commencement of the course, the results are shown in Table 5.

Table 5. Preliminary survey about the profile of the enrolled students in COMA.

35 years old students or older	43%
Active workers	50%
Undergraduate students in a related field	18%
Graduate or postgraduate students in a related field	19%
Students with a non-university degree in a related field	23%
Students enrolled in this MOOC especially because of the use of a remote laboratory	81%

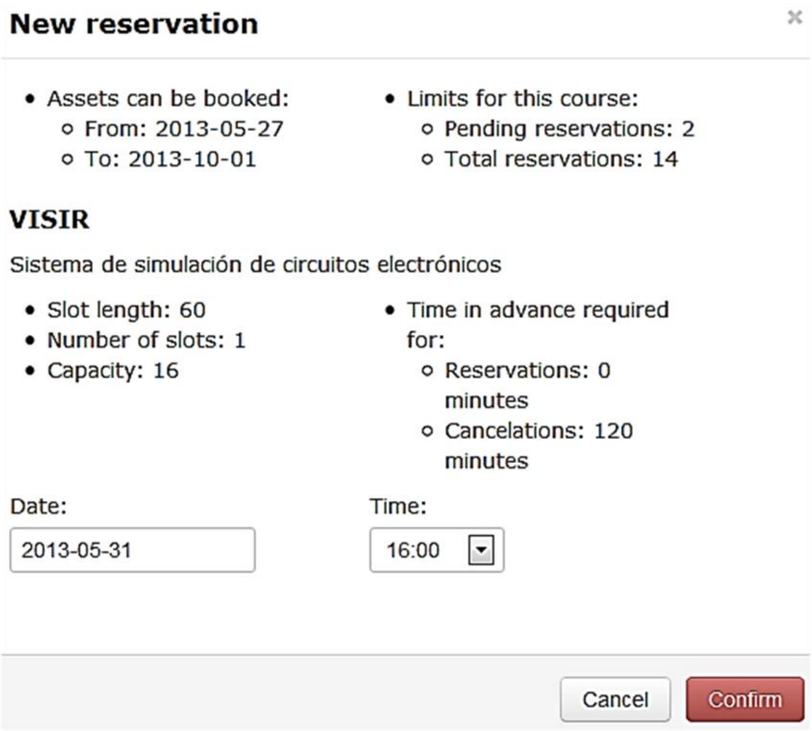
Access to experiments is provided by the MOOC's portal through an integrated scheduling system, as shown in Figure 25, which replaces the current scheduling system provided by VISIR, so that students can rely on a unique integrated educational portal. The initial settings allow 16 simultaneous users per 60 minutes slot and for each user a maximum of two simultaneous slots booked and a limitation of 14 slots per course. With these settings, VISIR allows up to 384 students to experiment with any of the designed practices of the MOOC every day.

3.5. Industrial Electronics Experiments[1]

Up till now, the prevalent remote laboratory solutions for electronic circuit's practices typically address I/O characteristics of a specific circuit connection. A further step was developing a first-of-its-kind set of remote industrial electronics experiments, oriented to labor markets and industrial needs in order to diminish the gap between academia and workplace. The experiments enable: studying the behavior of electronic components and commercial ICs; using manufacturers'

[1] In this section, I would like to acknowledge particularly the contribution of Prof. Santiago Monteso from UNED.

datasheets and comparing them with measured values; monitoring harmonic distortions—with a high precision—due non-linear impendency of circuits, as well as studying the role of I/O filters in mitigating these distortions; and calculating heat dissipation in electronic components either in transient or in steady state, as well as studying the effect of room temperature and of applied heat sinks in heat dissipation. The experiments were realized, also, taking into consideration issues such as safety and protection of components, configuration for high precision measurement with minimum possible distortion, and full switching and automation mechanism. Combining this kind of experiments with VISIR, as a result, yielded a unique training platform of its kind for remote industrial electronics experiments. The design patterns of the experiments' circuits are emphasized in this section without delving into theory details. From each experiment circuit numerous exercises are derived such as: comparing between simulations or theoretical calculations and measurement; and comparing between datasheets and component's behavior. The new designed circuits are shown in Figure 26.



New reservation

- Assets can be booked:
  - From: 2013-05-27
  - To: 2013-10-01
- Limits for this course:
  - Pending reservations: 2
  - Total reservations: 14

VISIR

Sistema de simulación de circuitos electrónicos

- Slot length: 60
- Number of slots: 1
- Capacity: 16
- Time in advance required for:
  - Reservations: 0 minutes
  - Cancelations: 120 minutes

Date: 2013-05-31

Time: 16:00

Figure 25. Reservation system in MOOC-UNED.

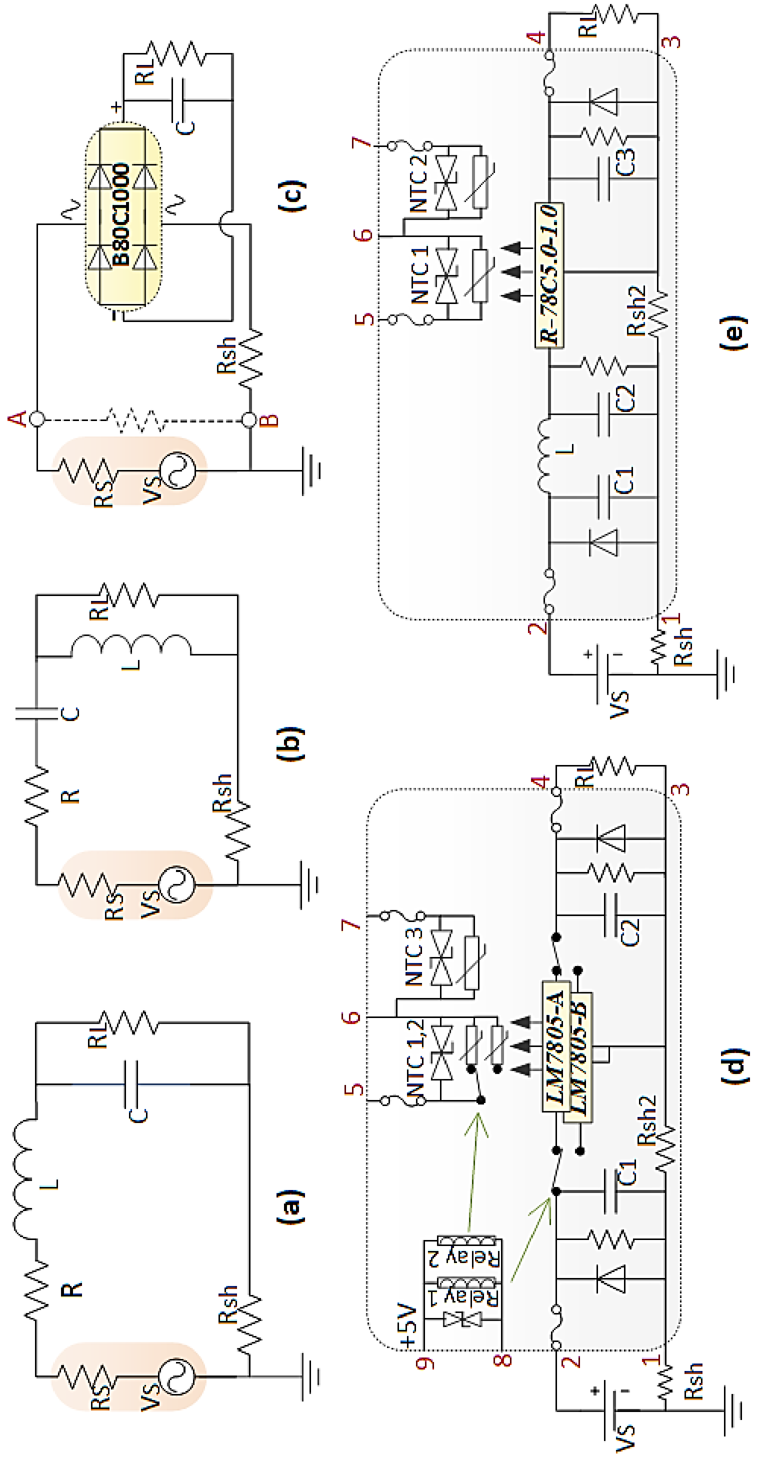


Figure 26. New designed industrial electronics circuits.

3.5.1. **Second-Order Low-Pass RLC Filter**

This circuit is shown in Figure 26(a). The purpose of the circuit is to study the effect of the filter on applied input signals with different frequencies. A semi-log scale of frequency versus the phase shift and versus the magnitude of the ratio of voltages in decibel (bode plot) is obtained to show the filter characteristics. Signals with frequencies lower than the resonance frequency are slightly altered and greatly attenuated otherwise. The phase shift angle is shifted from 0 º towards -180º as frequency increases. The slope in the frequency response curve of the second order circuits is steeper than that of first order circuits, which is recommended for better attenuation. The input signal is a square waveform and it is analyzed using Fourier series in order to study the effect of the filter on each harmonic sine wave separately with its own frequency. Finally, the filtered signal is monitored at resonance frequency and the amplification gain is calculated. The amplification gain is inversely proportional to the series resistance value.

The inductor is protected by a 1.5KE Transient Voltage-Suppression (TVS) diode in order to absorb transient voltage spikes. The operational voltage of the TVS diode is higher than that of the input signal, otherwise the current wouldn’t reach the conductor. The inductor was designed with a high saturation current, compared to the circuit’s current, to avoid saturation and to show ideal behavior. Thus, the filter’s effect could be clearly monitored.

A shunt resistor (Rsh) was added to each circuit in order to monitor with high precision the current on the oscilloscope. It was selected carefully with a rational value (approximately lower than 1% of the circuit’s resistance) and at the same time fairly high to avoid distortions as possible.

3.5.2. **Second-Order High-Pass RLC Filter**

The circuit is shown in Figure 26(b). It follows the same analysis procedures of the previous circuit but with a high-pass filter of a triangular input signal. Signals with frequencies lower than the cut off frequency are greatly attenuated and slightly altered otherwise. The phase shift angle is shifted from 180º towards 0º as frequency increases.

3.5.3. **AC/DC Converter (Full-Wave Rectifier)**

The circuit is shown in Figure 26 (c). The purpose of the circuit is to study the effect of the load and the output capacitive filter on the rectified output voltage, calculating the ripple and average value of the output voltage in each case. As well, to study the effect of the output capacitive filter on the distortion of the input signals. The full-wave rectifier passes positive swings in voltage and converts negative ones into positive swings at the output. The output frequency is twice the input frequency, and the average voltage at the output is 0.636 times the zero-to-peak output voltage. A B80C1000 bridge rectifier IC is used.

An output filter capacitor is connected to delay the discharge time and hence smooth out generated rectified voltage by diverting the AC portion of the input signal to ground, while passing the DC portion. The capacitor must be large enough to store a sufficient amount of energy to provide a steady supply of current to the load during the negative- cycle. If the capacitor is not large enough or is not being charged fast enough, the voltage will drop as the load demands more current. Despite the high tolerance of electrolytic capacitor (about ±5% to ±20%) they are used because of their high capacitance value. The electrolytic capacitors are protected by a diode to avoid reversing its polarity. As well, a resistor is connected across their terminals to ensure their discharge in a maximum time of 10 minutes. The maximum supported voltage of the capacitors is higher than the amplitude of the input signal and the current is adjusted by the power supply to a value accepted by the capacitors and the diode.

In order to monitor the distortion in the input voltage due to the non-linear impedance of the circuit, a high resistive load was added across A B—so that it will not affect the circuit's operation. A difficulty arises when it comes to monitoring voltage across load since the oscilloscope of VISIR is internally connected to ground. In this case we measure the maximum and minimum values of the output voltage by subtracting measurements of VD-to-ground from VC-to-ground at maximum and min values. The maximum value occurs when the input current reaches zero (when the capacitor starts to discharge) and the minimum value occurred when it moves from zero (when the capacitor starts to charge). The input current is monitored across Rsh.

3.5.4. Non-Isolated Linear Regulated DC/DC Converter

The circuit is shown in Figure 26(d). The purpose of the circuit is to study the effect of load variation on the input and output signals, the effect of input signals' variation on the output signals, and the thermal effect of the regulator IC due to power dissipation. The circuit controls the amount of current flowing through the load—so as to maintain a constant output voltage—by comparing the supply's DC output with a fixed internal reference voltage. A LM7805 linear regulator IC is used. A LC filter circuit is inserted to reduce voltage spikes and eliminate the emission of Radio Frequency Interference (RFI) by the power supply. The effect of the filter on the internal current is monitored across Rsh2 by connecting the oscilloscope to terminal 3.

The input and the output of the IC are protected against reverse polarity and excessive voltage by a 1.5KE unidirectional TVS. Capacitors are attached to the input and the output, as well, to eliminate spikes and maintain a constant output voltage with load variations. High resistance resistors are added for discharging capacitors without affecting circuit's operation. The current maximum value is adjusted by the power supply and extra-protection is provided by adding fuses to the input and output terminals of the entire circuit.

The major disadvantage of linear regulators is that the extra energy is dissipated as heat which increases temperature and reduces efficiency (Pout/Pin) to lower than 50% typically. Two regulator ICs are used, one with a heat sink (LM7805-A) and the other without (LM7805-B). In order to measure the temperature of the regulator IC in each case and see the effect of the heat sink in heat dissipation, a B57861S0103F045 NTC thermistor is used for each and a third one (NTC 3) is used to measure room temperature.

Two 30.22 electrical relays of nominal voltage 5V are used to switch connection between the two regulator ICs and their coils are excited by the same DC power source of VISIR (at open circuit they connects LM7805-A). The relays are protected by a 1.5KE unidirectional TVS of 6.8V. The datasheet of the thermistors is used to calculate the temperature and a reasonable time should be considered (about 5 to 10 minutes) between each measurement.

Users by error could apply voltage across thermistors connecting it to the power supply, which may destroy the thermistors. Therefore, each thermistor is protected by a 1.5KE bidirectional TVS of 15V and with a negligible surge current (1mA approximately) so that it doesn’t affect the measurements. Thus, the maximum allowed power dissipation across each thermistor is 60mW considering a worst case where temperature reaches 45° (regarding to the datasheet the resistance of the thermistor at this temperature is 4.369KΩ) owing to a previous utilization of the component. Additionally, a fuse was added to protect the TVS by controlling current passing to it.

3.5.5. Non-Isolated Switching Regulated DC/DC Converter

The circuit is shown in Figure 26(e). It has the same purpose of the previous circuit but with an R-78C5.0-1.0 switching regulator instead. The regulator is step-down configured (buck converter) and thus its output voltage is smaller than its input voltage. Switching regulators are much more efficient than their linear counterpart; their power efficiency can exceed 85%. The overall connection of the designed circuits in the matrix is shown in Figure 27.

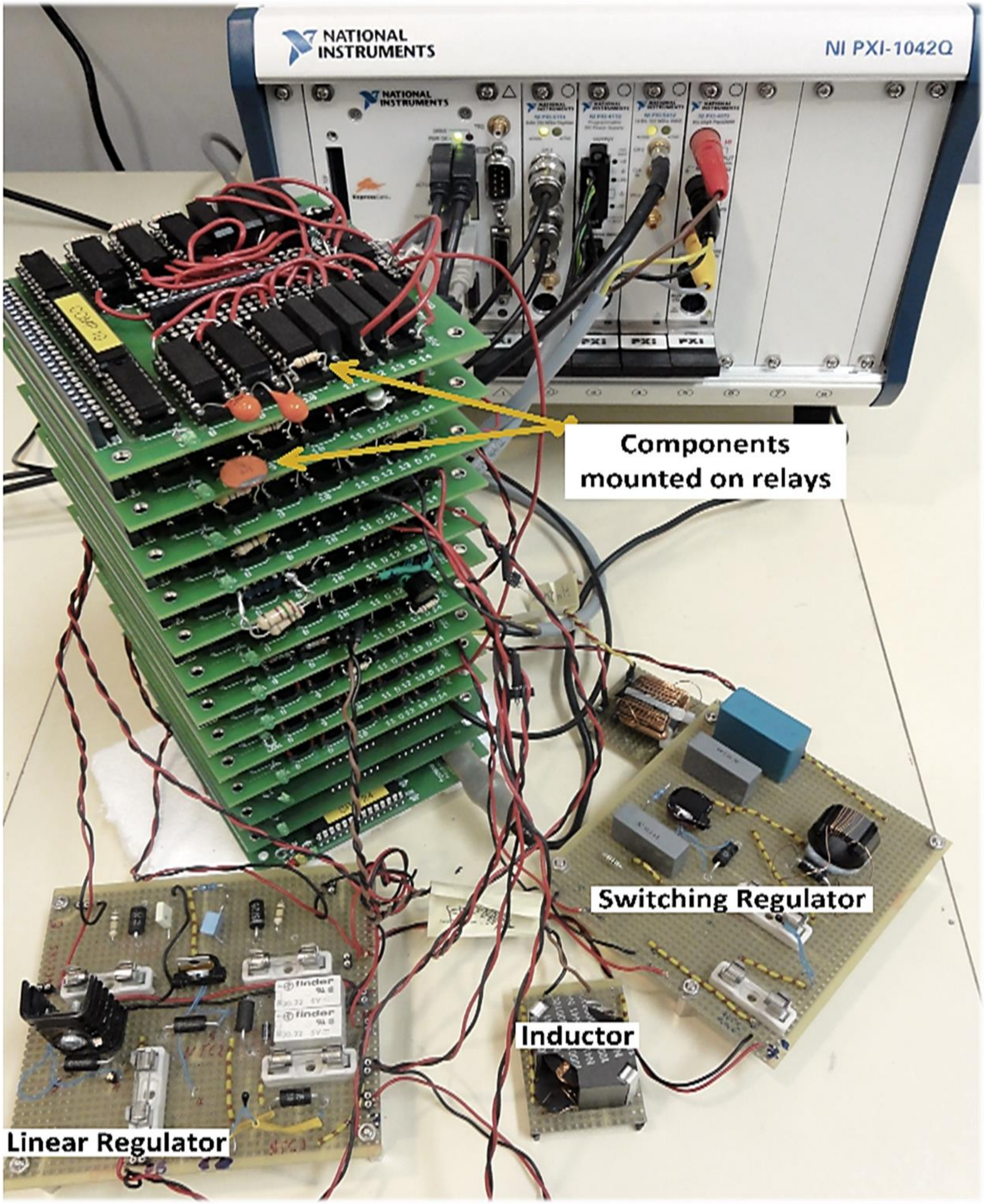


Figure 27. Designed circuits connected to the matrix.

3.5.6. **Remote Retrieved Results**

As a final phase, the developed circuits were tested. The circuits were mounted and wired remotely in the virtual workbench as shown in Figure 28.

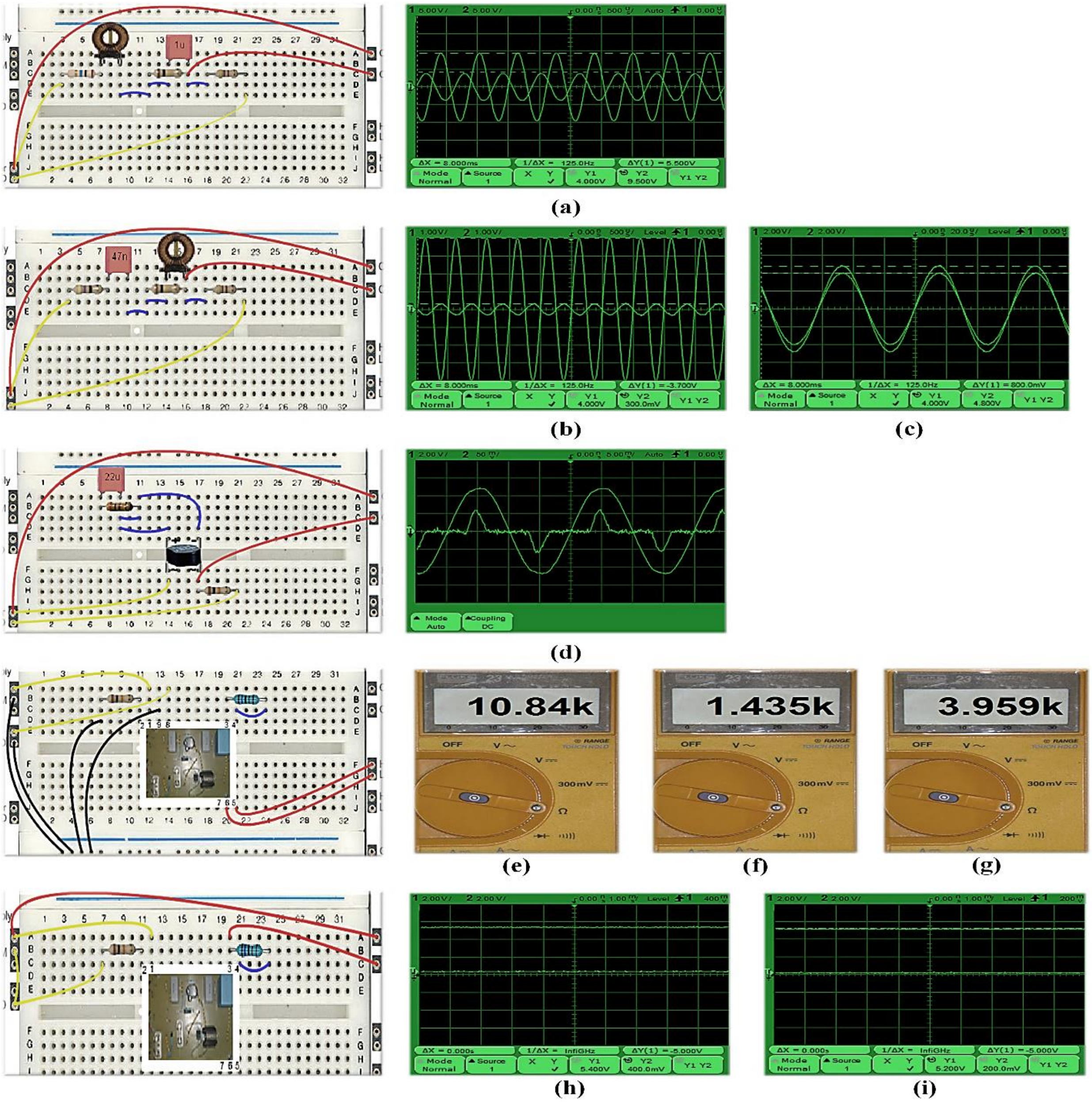


Figure 28. Online retrieved measurements form the new designed experiments.

Afterwards, online measurement results are retrieved from each circuit as a case study. The necessary configurations in VISIR and the datasheets are provided in Appendix B.

The selected measurements are the following:

1) The circuit shown in Figure 26(a) is mounted with the following values: R= 13.5 Ω, RL= 100 kΩ, C= 1 µF, L= 9.9 mH, and VS= 8Vpp sinusoidal. The output voltage across RL versus the input voltage, at resonance frequency, is monitored as shown in Figure 28(a). The resonance frequency is calculated in (1) and the output voltage is amplified with a gain calculated in (2).

$$f_{resonance} = \frac{1}{2\pi\sqrt{LC}} = 1592\,Hz \qquad (1)$$

$$\left|\frac{VL}{Vs}\right|_{peak} = \left(\frac{9.5}{4}\right) \times 100 = 237.5\% \qquad (2)$$

2) The circuit shown in Figure 26(b) is mounted with the following values: R= 13.5Ω, RL= 100 kΩ, C= 47 nF, L= 9, 9 mH, and Vs= 8 Vpp sinusoidal. The output voltage across RL versus the input voltage, at frequencies 4 kHz and 16 kHz is monitored as shown in Figure 28(b) and Figure 28(c) respectively. Signals with frequencies lower than resonance frequency (7324 Hz) are greatly attenuated and slightly altered otherwise.

3) The circuit shown in Figure 26(c) is mounted with the following values: R= 3 kΩ, Rsh= 10 Ω, Vs= 20 Vpp sinusoidal (f= 50 Hz), and C= 22 µF (electrolytic 16V). The input current measured across Rsh versus the input voltage is monitored in Figure 28(d). The current curve shows the effect of the bridge rectifier along with some distortions owing to the low value of the shunt resistor.

4) The circuit shown in Figure 26(d) is mounted with the following values: RL= 75 Ω, Vs= 25 V, and Vrelays= 5V DC (in case we want to connect LM7805-B otherwise it is not connected). While the following values are fixed values in the design of the black box: C1= 0.33 µF, C2= 0.1 µF, Rsh2= 10 Ω, and NTC1= NTC2= NTC3= 10 kΩ (25°C, 1%). The datasheet of the NTCs and the linear regulator are consulted in the following measurements. First, the temperature of the ambient is measured by the NTC3 as shown in Figure 28(e), which corresponds to approximately 25°C. Afterwards, the input signal is applied to the regulator LM7805-A for 5 minutes and then its temperature is measured by the NTC1 as shown in Figure 28 (f), which corresponds to approximately 75°C. By subtracting the room temperature, the increment in regulator's temperature will be 50°C. Theoretically, the power loss or the power dissipated across the regulator could be approximated using (3) and neglecting the protection components.

$$P_{loss} = V_{reg} \times I_L = \left(V_S - V_L\right) \times \left(\frac{V_L}{R_L}\right) = \left(20-5\right) \times \left(\frac{5}{75}\right) = 1W \qquad (3)$$

If the thermal resistance of the regulator equals 60 °C/W (junction air thermal resistance – junction cases thermal resistance), thus in ideal case its temperature at steady state should be 60°C for a unit power dissipation, which is nearly close to the practical measurement—considering measurement drops between the physical component and sensor and the tolerance of the materials. Afterwards, the regulator LM7805-B is connected by applying a 5V DC input signal to the relays. The temperature of the regulator LM7805-B is measured by the NTC2 as shown in Figure 28(g), which corresponds to an increase of approximately 50°C. By subtracting the room temperature, the increment in regulator's temperature will be 50°. The total thermal resistance is 21°C/W, which is the sum of the thermal resistances of the heat sink (14 °C/W) and the junction cases of the regulator, in addition to 2 °C/W due to the imperfection of the contact between the regulator and the heat sink. Thus theoretically, the in ideal case its temperature at steady state should be 21°C for a unit power dissipation, which approximates with the practical measurement.

5) The circuit shown in Figure 26(e) is mounted with the following values: RL= 75 Ω and Vs= 25 V. While the following values are fixed values in the design of the black box: C1= 22 μF, C2= 10 μF, C3= 10 μF, L= 56 μH, Rsh2= 10 Ω, and NTC1= NTC2= 10 kΩ (25º, 1%). Figure 28(h) and Figure 28(i) show the output voltage (difference of voltages measured at points 3 and 4) at applied voltage of 20 V and 10 V respectively, which is fixed to 5 V regardless of the applied voltage.

3.6. Implementation in a Master Degree Program

The new designed practices are delivered within an online official and inter-institutional master's degree program in ICT. It is conducted by 5 European institutions from 4 different European countries and oriented to labor market needs. The program is funded by the European project, RIPLECS [47] and is targeted to engineers, technicians and scientists with interest on up-to-date topics in the area. Students acquire skills focused on industrial field like production organization, design of products, processes and installations, quality management or team management. Thus, part of the project management includes contacting and collaboration with enterprises in the sector to respond to the labor market needs. The master's program relies on a distributed and adaptive LMS—developed by the project partners—enabling world-wide distribution of learning resources and remote lab-experiments, by utilizing multiple Web servers across each partner's university within a single network topology. Instructors from each university can take the advantages of conducting the program and deploying remote lab experiments in their native language and personal educational point of view [48, 49]. The project partners are the following Institutions:

- Electrical & Computer Engineering Department, UNED, Spain.
- Communication & Control Systems Department, UNED, Spain.
- University of Plovdiv (PU), Bulgaria.
- Cork Institute of Technology (DEIS), Ireland.
- Technical University of Sofia (TUS), Bulgaria.

- Institute for Technical Informatics, Graz University of Technology (TUGraz), Austria.

The program was launched in the academic year 2013/2014 and will be delivered as many years as demands last. The program is based on European Credit Transfer and Accumulation System (ECTS) and is officially-accredited initially in Spain, by the Spanish ANECA accreditation Agency (www.aneca.es). In addition, a certification from all partner universities are included. The program is of one year (2 semesters) and 60 ECT, and is composed of three modules; fundamental module, specialized module, and final project module. The subjects are of 5 ECT, and the final project is of 10 ECT. All subjects are taught in English and any student around the world can be registered and enrolled in the master. The program's curriculum is organized as shown in Table 6.

Table 6. Curriculum of the RIPLECS master's degree program.

		Subject	**Provider**
Semester 1 (Fundamentals) October-February		▪Introduction to Information and Telecommunication systems	PU
		▪Industrial and Real-time Communications	UNED
		▪Internet Technology	DEIS
		▪Electronics for Information and Communication Technologies	TUS
		▪ICTs research and engineering competence skills	UNED
Semester 2 (Specialized) February-June		▪Microprocessor Techniques	PU & UNED
		▪Wireless Communications	PU & TUGraz
		▪Multi Media	DEIS
	Two of electives	▪Power Supplies for ICT Equipment	UNED
		▪Microelectronics	PU
		▪Satellite and Mobile Communications	PU & UNED
		▪Computer Modeling and Simulation of Electronic Circuits	PU & UNED
		▪Final Project	

The new designed practices are delivered as a core module within the subject "*Power Supplies for ICT Equipment*" and access to the practices is directly provided from the program's LMS.

3.7. **Summary**

In this chapter, development and implementation of two sets of electronics experiments using VISIR at UNED were reported. The first set is targeted to the undergraduate engineering curricula and it was deployed in official undergraduate engineering degrees at UNED. As well, in a Massive Open Online Course (MOOC) delivered by UNED to the public. The other is a set of new advanced remote electronics experiments oriented to postgraduates and apprentices and addresses labor markets and industrial needs in order to diminish the gap between academia and workplace. Industrial-related issues were emphasized in the design to allow understanding the behavior of electronics components. This kind of experiments hadn't been previously reported in the literature either in conventional lab or remote lab arrangement. This combination, as a result, converted VISIR system into a unique training platform of its kind for remote electronics experiments. The experiments enabled: studying the behavior of electronics components and commercial ICs; using manufacturers' datasheets and comparing them with measured values; monitoring harmonic distortions—with a high precision—due non-linear impendence of circuits, as well as studying the role of I/O filters in mitigating these distortions; and calculating heat dissipation in electronic components either in transient or in steady state, as well as studying the effect of room temperature and of applied heat sinks in heat dissipation. Afterwards, the system was tested and measurement results were retrieved remotely from the mounted circuits are provided as a case study. Finally, the chapter addressed the implementation of these experiments in an inter-institutional European online master degree program in ICS.

This page intentionally left blank

Chapter 4

4. Integration into Educational Systems

The vast majority of the remote laboratories we have come across in the last two sections were developed with their own administration portal, which provides implementation services such as scheduling and profile roles. For students who need to access several remote laboratories from different providers, using different portals and being required to login in each would be inconvenient. Recent developments tends to integrate remote laboratories into existing educational system in lieu of building a custom portal for each lab [50]. This practice is far more efficient for the following reasons:

1. Student could access to a set of remote laboratories from different providers from a unique portal and using the same credentials.
2. Indexing different kinds of remote laboratories from different providers and from different applications could be easier for further searchers.
3. This enables providing an integral environment that combines remote laboratories with other learning resources designed for a specific course.

Therefore, integration into educational systems is a crucial stage factor in the design of today's remote laboratories. Integration scenarios of remote laboratories with heterogeneous educational systems could be classified as follows.

4.1. Integration into Metadata Repositories

Commonly, a remote laboratory of a university is scarcely reused by other universities due to the lack of information about the laboratory. The Lab2go project (www.lab2go.net/) [51] was launched to fill this gap. It is a web portal, inspired from the semantic Web feature of Web 3.0, which acts as a repository and provides a common framework for online laboratories providers all over the world. The laboratories with all their related information are added with metadata by using semantic web technologies such as: Resource Description Framework (RDF) for expressing data models, merging different data sets, and employing a class hierarchy to represent commonality and variability; and Web Ontology Language (OWL) for adding relationships based on ontology to the classes and properties. This allows expressing a piece of data about some entity in a way that it can be combined with information from other sources to facilitate their allocation and precisely define the searching criteria rather than the traditional available searching tools that are based on keywords. Several widely accepted and well-supported standards were used to define the ontology and to simplify the mapping between different models. Basic terminology and data types for describing resources and learning objects were adopted from Dublin Core and Learning Object Metadata (LOM). Friend of a Friend (FOAF) ontology was used to describe relations between people, their organizations and the projects on which they are working. The ontology is structured as shown in Figure 29.

Two main classes are defined: online laboratory and experiment. Each class has its own priorities and subclasses. Each of the three subclasses of the online laboratory class can be linked to one or more experiment class. The ontology was designed using the software Protégé (protege.stanford.edu) and the portal was developed by the semantic Wiki OntoWiki (ontowiki.net)—which enables allows users to manage structured data extended with some Wiki-like functions that permit collaborative editing of content. Lab2go, nevertheless, doesn’t provide access to remote laboratories.

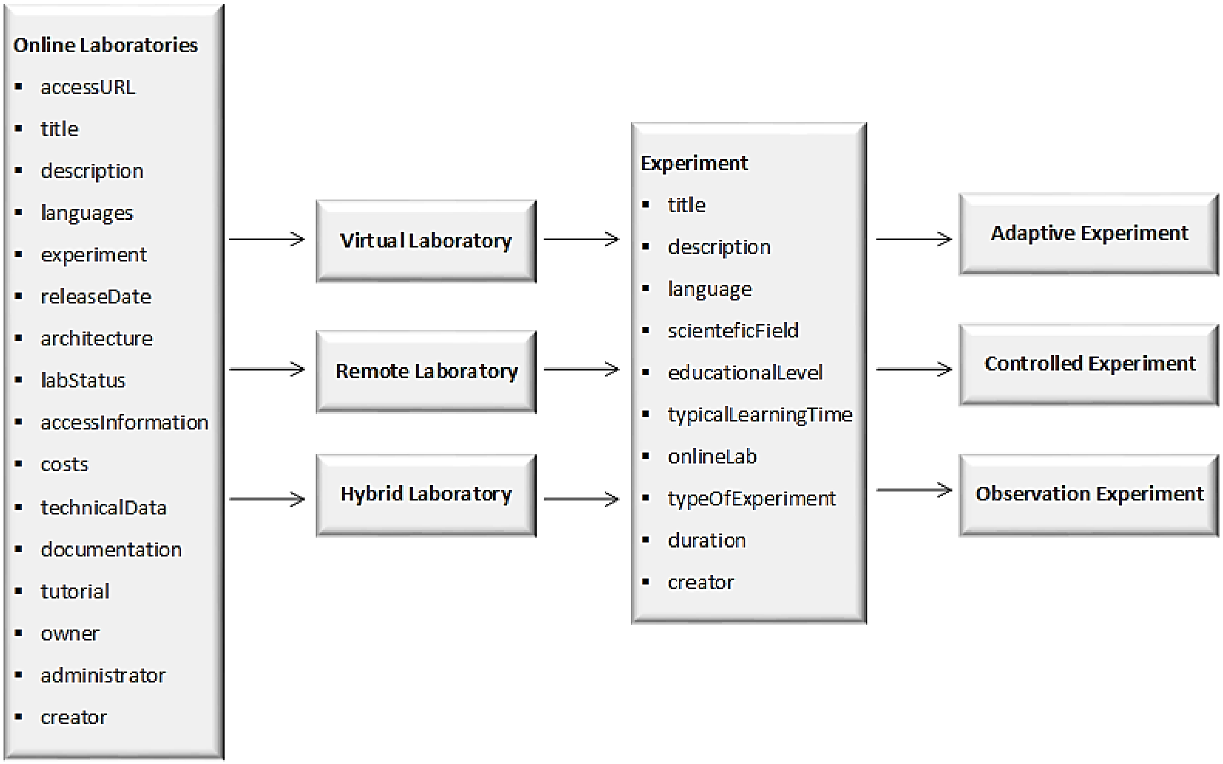


Figure 29. Class structure of Lab2go ontology [51] .

4.2. Integration into Learning Management Systems (LMSs)

LMSs have been widely used in distance education for the past two decades. LMS is a software application for the realizing classrooms and courses online. They provide many tools and features such as:

- **Administration tools:** user registration, account roles, user profile, assign tutors, students and groups, billings, design course contents, scheduling, etc.
- **Synchronous and asynchronous communication tools:** chat, forums, video conference, webinars, events, news, emails, calendars, blogs etc.
- **Multimedia sharing tools:** upload and download videos, audios, photos, files, etc.
- **Evaluation and tracking tool:** surveys, exams, assignments, user tracking, etc.

- **Standard compatibility:** LMS organizes the content in a hierarchical structure with regarding to a specific standard in order to allow swapping contents between different LMS without re-writing it again. From the most common used standards are: Shareable Courseware Object Reference Model (SCORM) & IMS for content packing, IMS Question and Test Interoperability (QTI) for tests and evaluations, LOM and Dublin Core for describing and reusing learning objects.

LMSs can be open source such as: Moodle (moodle.org), dotLRN (openacs.org/projects/dotlrn/), Sakai (www.sakaiproject.org), Claroline (www.claroline.net), etc. which could be easily developed and redesigned, thus all our researches are realized focusing on these types of LMS. While Proprietary types such as Blackboard (www.blackboard.com), JoomlaLMS (www.joomlalms.com), SharePointLMS (www.sharepointlms.com), etc. could only be modified by their developers. Even though, most of the features provided by LMSs are of crucial importance to practical sessions; LMSs are confined to theoretical resources and doesn't support their practical counterparts. In this vein, several initiatives was launched for integrating remote laboratories into open source LMSs. The goal was to make use of all the services provided by LMSs and apply them in the remote practical lab sessions as shown in Figure 30.

A significant approach in this regards was realized in the LiLa project (www.lila-project.org) [52], which aims to wrap remote laboratories interfaces along with other scaffolding and learning resources in SCORM packages as a Sharable Content Objects (SCOs) or LiLa Learning Objects (LLOs) insofar that they can be rendered in either SCORM-compliant LMSs or in the project's portal, LiLa portal (www.library-of-labs.org). Contents are packaged in a .ZIP file and described in a XML manifest file, which contains Dublin Core and/or LOM based metadata in addition to a URL to the LiLa booking system server. Shibboleth was adopted for authentication and authorization in the LiLa portal and can be exported with LLOs to LMSs. A JavaScript code is embedded into LLO to check with the booking system through Web services calls if the student has a valid reservation. If this is not the case, the code will render an informative message and will redirect the user to an interface in the booking system server to allow him/her making a new reservation. The LiLa portal includes an extra layer represented by the 3D virtual world engine, Open Wonderland (openwonderland.org), as a collaboration environment for students, teachers and researchers. In [53, 54], another important approach is found, in which a generic Web-services based middleware was architecture was developed for integrating remote laboratories with open source LMSs such as Moodle and DotLRN.

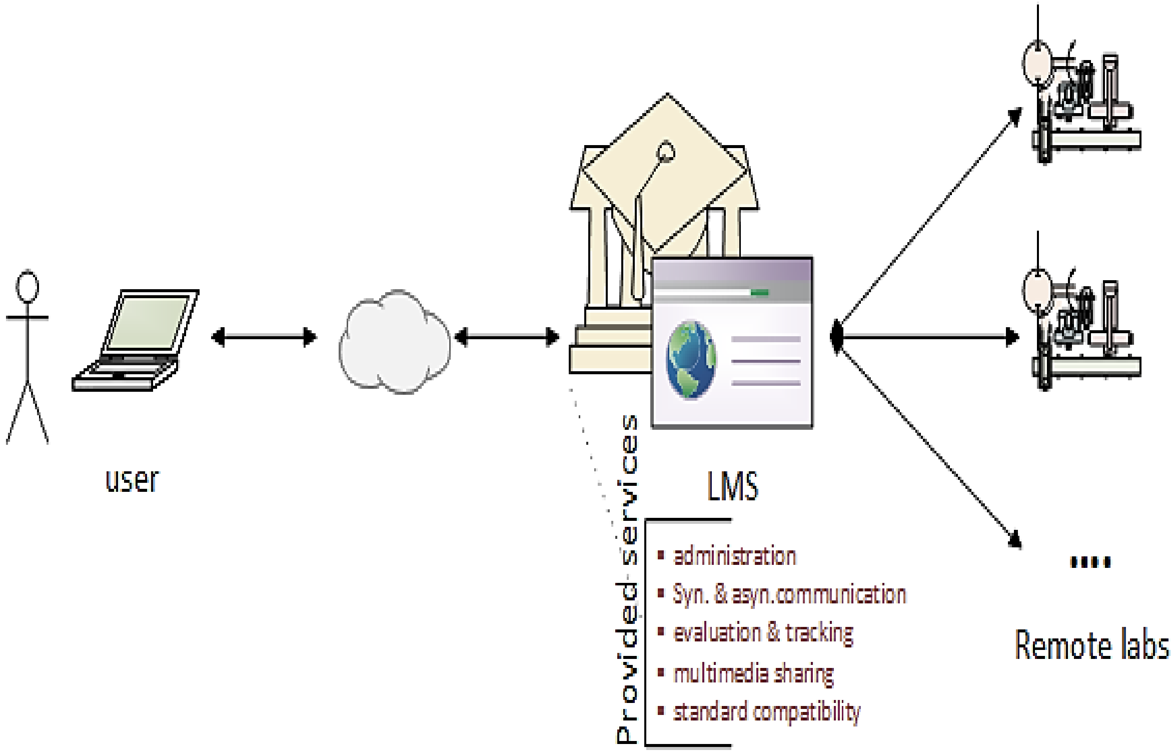


Figure 30. Integrating remote laboratories into LMSs.

4.3. Integration into Personal Learning Environments (PLEs)

PLEs allow learners to build and customize their own learning environment with a full control of their learning process in an informal context that is neither tied to university nor corporation. Unlike LMSs, PLEs don't support compatibility and don't provide full students-teacher support in order to enable delivering classrooms online. Instead they embrace Web 2.0 and the mashup concept to support personalization, which is not supported by concurrent LMSs. Learners use PLEs to import external loosely-coupled services and social media from Web 2.0 and mash them up in their own space. Aggregated services (e.g., remote laboratories, widgets, online videos, etc.) are accesses from the PLE portal while being hosted at an external server as demonstrated in Figure 31.

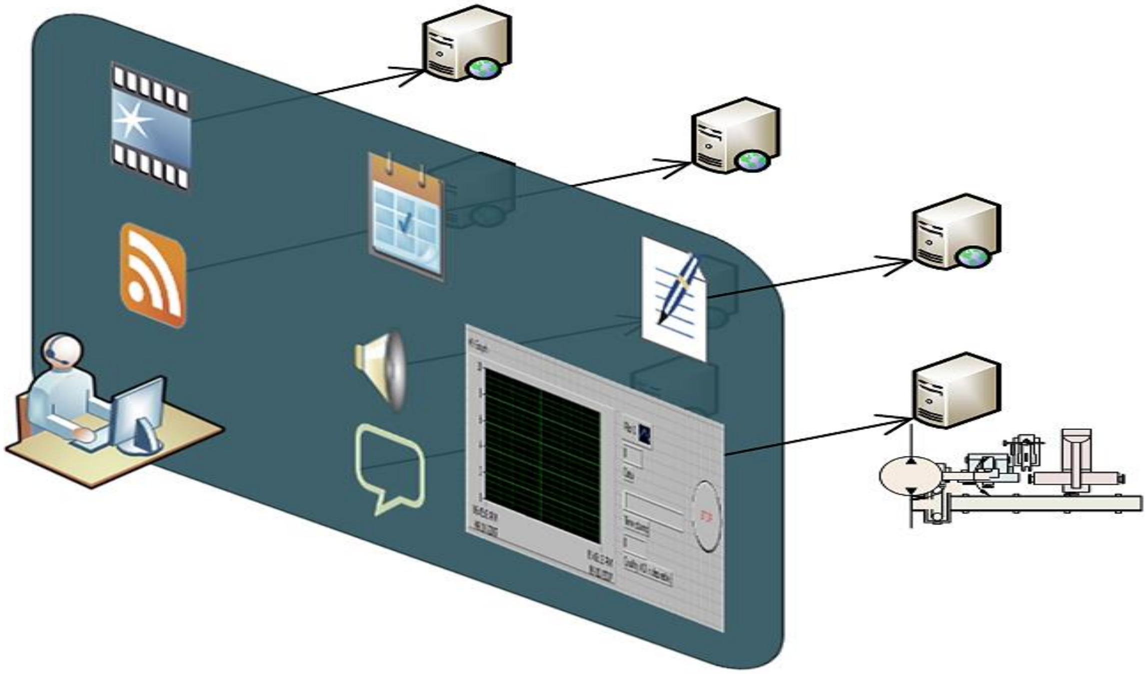


Figure 31. Personal Learning Environment (PLE).

Graaasp (graasp.epfl.ch) [55] is a well-known PLE in which remote laboratories have been usefully integrated. Graaasp has four types of entities, which are arranged dynamically to allow sharing resources and creating a customized environment: resources (e.g., wikis, YouTube videos, etc.), applications (e.g., widgets or applets), spaces (e.g., group activities, lessons, forums, etc.), and people (users). In [56], remote laboratories are divided into modular components or widgets and aggregated dynamically to Graaasp, which acts as a widget engine or container. Widgets were created using JavaScript and Dynamic HTML (DHTML) and rely on the widget stander OpenSocial (http://opensocial.org/). They don't require any plug-in to operate and they would work in any OpenSocial-complaint container.

4.4. Integration into Remote Laboratory Management Systems (RLMSs)

A RLMS is a Web application that provides a common online portal for accessing and administrating a wide pool of heterogeneous remote lab systems that might be distributed across different institutions and geographical locations as shown in Figure 32.

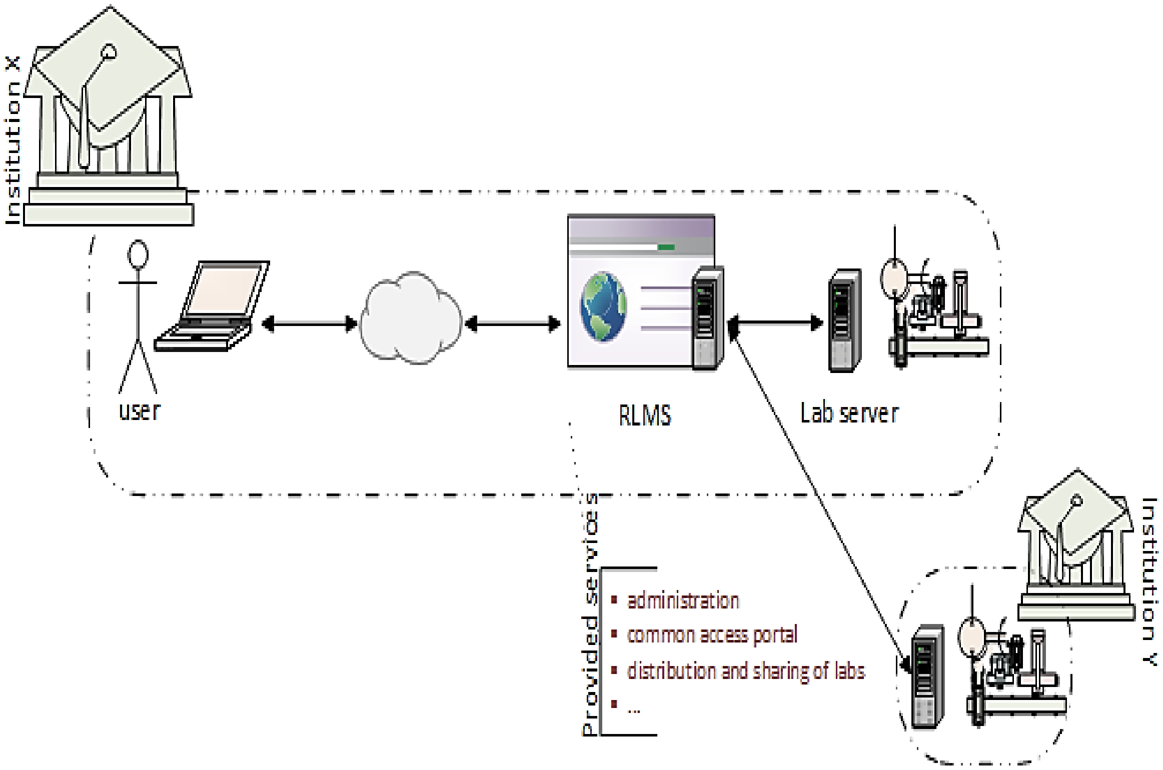


Figure 32. Architecture of RLMS.

As demonstrated in Figure 19, students at institution X could access through the RLMS of their institution to a wide poles of remote laboratories installed at their institution and at other institutions as well (e.g., institution Y) through a unique online portal and with the same credentials. The server of the RLMS handles the internal connections with other remote laboratory servers that might be located at other institutions. RLMSs should be agnostic with regard to the remote laboratory design in order to support the widest range possible of remote laboratories. It is claimed that this can lead to improved utilization levels, shared resources and costs, and access to a much broader range of laboratory apparatus. Prominent RLMSs, which have been successfully implemented across many universities, includes iLab Shared Architecture (ISA) [57] and Sahara [58]. However, there exist other remarkable RLMSs such as WebLab Deusto [59].

ISA is a middleware infrastructure, based on Web services, that is developed within the iLab project [60] at the MIT. ISA is an efficient management framework that can support administration and access to a wide variety of platform-independent online laboratories. The architecture started with a three-tiered model, consisting of lab clients, service broker middleware, and lab servers. The service broker is responsible for providing generic functionalities such as authorization, scheduling, data storage, etc. It is typically located at the client side campus and it is able to be connected to multiple lab servers at distinct institutions with Web services. Conversely, a given lab server can receive experiments from an arbitrary number of service brokers. The lab server then serves these requests using a simple queue. This initial architecture was only valid for batched experiments. For these experiments the user specifies the experiment in the client which passes the experiment specification to the service broker, which then passes the specification to the lab server to be executed. The student's experiment specifications are then queued and, once executed, the results are returned back to the service broker to be provided to the user. The student doesn't have any connection to the lab server and indeed does not need to be online while his/her experiment is being executed.

Interactive experiments differ from their batched counterparts as they require control of lab hardware while the user sets parameters and observes results during the experiment execution. This requires that the user be able to schedule access, and that lab client be able to communicate with the lab server during experiment execution. In order to accommodate these requirements, ISA was extended to include a Lab Side Scheduling Service (LSS), a User Side Scheduling Service (USS), an Experiment Storage Service (ESS), and support for Ticketing and high bandwidth communication between the lab client and the lab server as demonstrated in Figure 33.

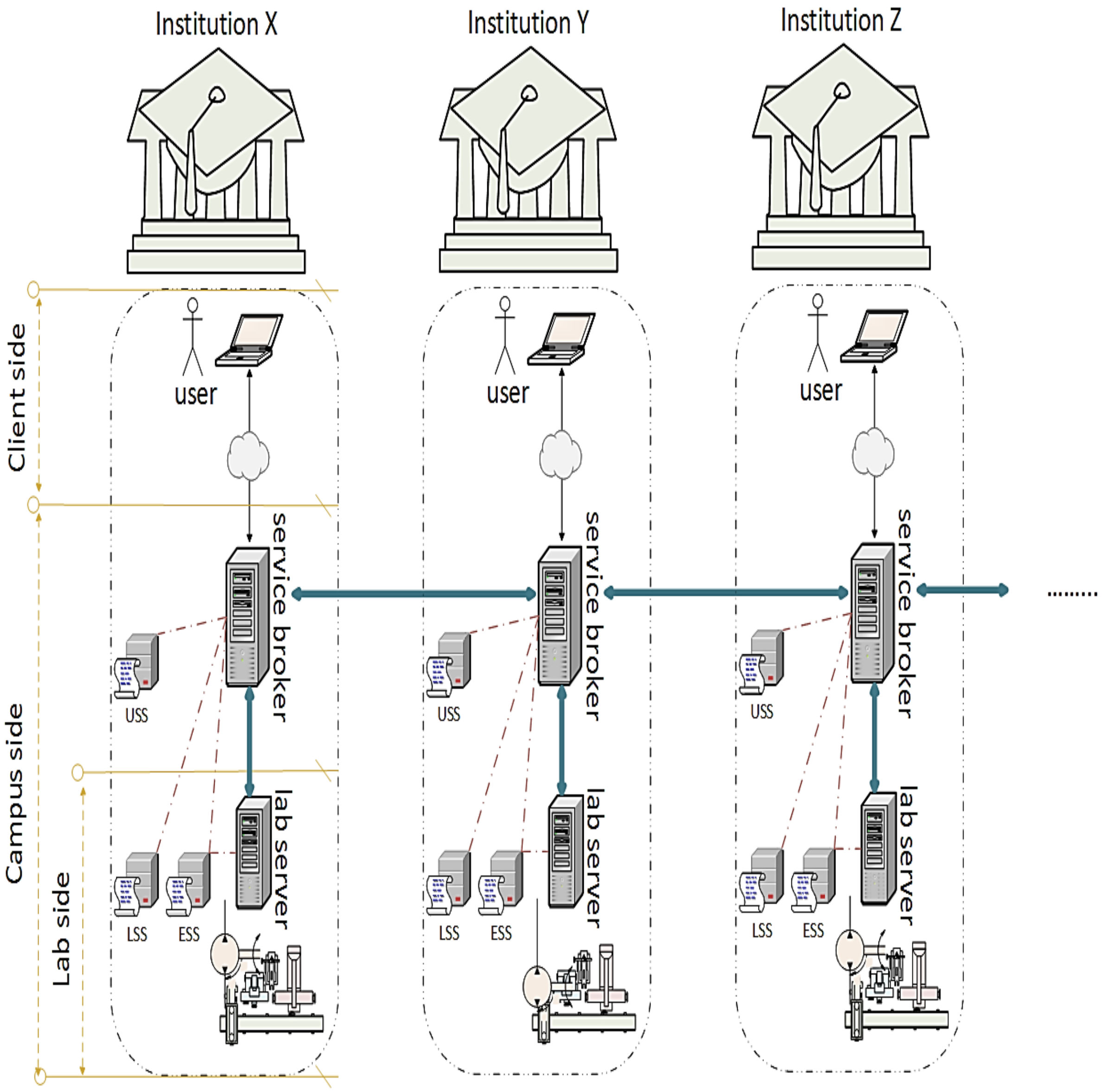


Figure 33. ISA for interactive experiment.

As demonstrated in Figure 33, student at institution X access with his/her credentials to the service broker installed at his/her institution. Once he/she is logged in, he/she could accesses laboratories available either at his own campus or at other campus of another institution (e.g., institution Y, or Z). This is achieved thanks to the intercommunication between service brokers of each institution (i.e., which is marked by a blue line). The LSS, USS, and ESS of each institution are implemented as Web services. The USS and LSS components set the policy of the student institution and the laboratory, respectively with regard to the scheduling of access, and are designed to work together. Storage of results is supported by the ESS for interactive experiments. The service broker is still responsible for authentication and authorization, but in this case, to gain access, a user authenticates with the service broker which provides the client with a ticket. This ticket is then presented to the lab server and to the ESS by the client enabling client access and control of the experiment and, at the same time, stepping out of the service broker.

Sahara, on the other hand, is a RLMS developed and adopted within the Labshare project [61], which was leaded by UTS [62]. Sahara allows institutions to use a common sharing platform for access to their experiments without the need for proprietary client software. The following terms are used by Sahara and are important for the subsequent discussions:

- **Rig instance (rig):** a single instance of a physical system made from various devices (including associated software and hardware) that can be used by one or more students in carrying out a distinct and discrete learning activity.
- **Rig type:** the class to which a rig belongs, and within which any rig can be used interchangeably. A given institution (or laboratory provider) will often have more than one rig type, and for each rig type they may have multiple rigs.
- **Rig set (pool):** the collection of homogeneous rigs, all of the same rig type, which may be distributed across multiple physical locations and managed by different organizations, but which can be treated as a single logical collection for access purposes.
- **Rig capability:** a list of identifying tags which may be used to correlate rigs into collections to queue or book for. This allows, for example, multiple rig types to be collectively queued to get the first free rig in any of the rig types.

The Sahara architecture is composed of three main software applications and each application is composed from a set of components. The components of the three applications intercommunicate with each other by Web services. Next, a brief description of each application is presented:

1) **Rig client:** the rig client provides a software abstraction of a rig and it is the intermediary between the scheduling server and the physical hardware (the rig itself). Each rig has its own rig client. It is typically (though not necessarily) written in Java and usually resides on the same server as the hardware interfaces to the rig. It allows a rig provider with administrative tools to control and configure the rig, including aspects such as testing, assigning a rig IP, listening port, and rig type, and registering the corresponding Webcam. Once the rig provider configures the rig, he/she puts it online to be registered in the scheduling server.
2) **Scheduling server:** the scheduling server is the heart of Sahara. It is written in Java and it carries out all the administrative tasks such as creating user classes, users, and user access keys, and registering all the rigs. The administrator configures the class details and assigns users, rig types or capability, and time reservation slots to that class. The arbitrator software component included in the scheduling server is responsible of allocating either the available registered rig from a rig type or a specific rig upon user request.
3) **Web interface:** the Web interface is the end-user portal for Sahara through which users are authenticated and log in, if they are assigned to a class. It is written in PHP, HTML and JavaScript.

Additionally, a database (e.g., MySQL or PostgreSQL) is needed and it communicates with the Web interface and the scheduling server. At UTS, all the rigs are installed in separate virtual machines, which are hosted in the same master server. The scheduling server that carries out the administration process of all the rigs is installed in a separate server. The Web interface could either be installed in the same server of the scheduling server or separately. The virtualization software, VMware (www.vmware.com), is used to set up multiple virtual PCs on a laboratory master server. The necessary software for each rig is installed in a single virtual machine. An arbitrator software system boots a Windows virtual machine on the master server and associates it with the relevant rig. The arbitrator authenticates requests for equipment and then allocates rigs to students from the unused rig pool, queuing the allocation request when all rigs are busy. The student connects to this virtual machine through Remote Desktop Protocol (RDP), runs the control application, and controls the rig. The control application is therefore running on the virtual machine at the master server (not on the remote user's computer). When a session of use is completed by a student, the arbitrator reclaims the rig, reinitializes it, and returns the rig to the free pool. Additionally, and instead of using a remote desktop connection, a rig UI can be built separately, by using Web standards and/or any programming language in order to communicate with the rig control application that runs on a virtual machine, and embedded within the Sahara Web interface. The overall architecture is depicted in Figure 34.

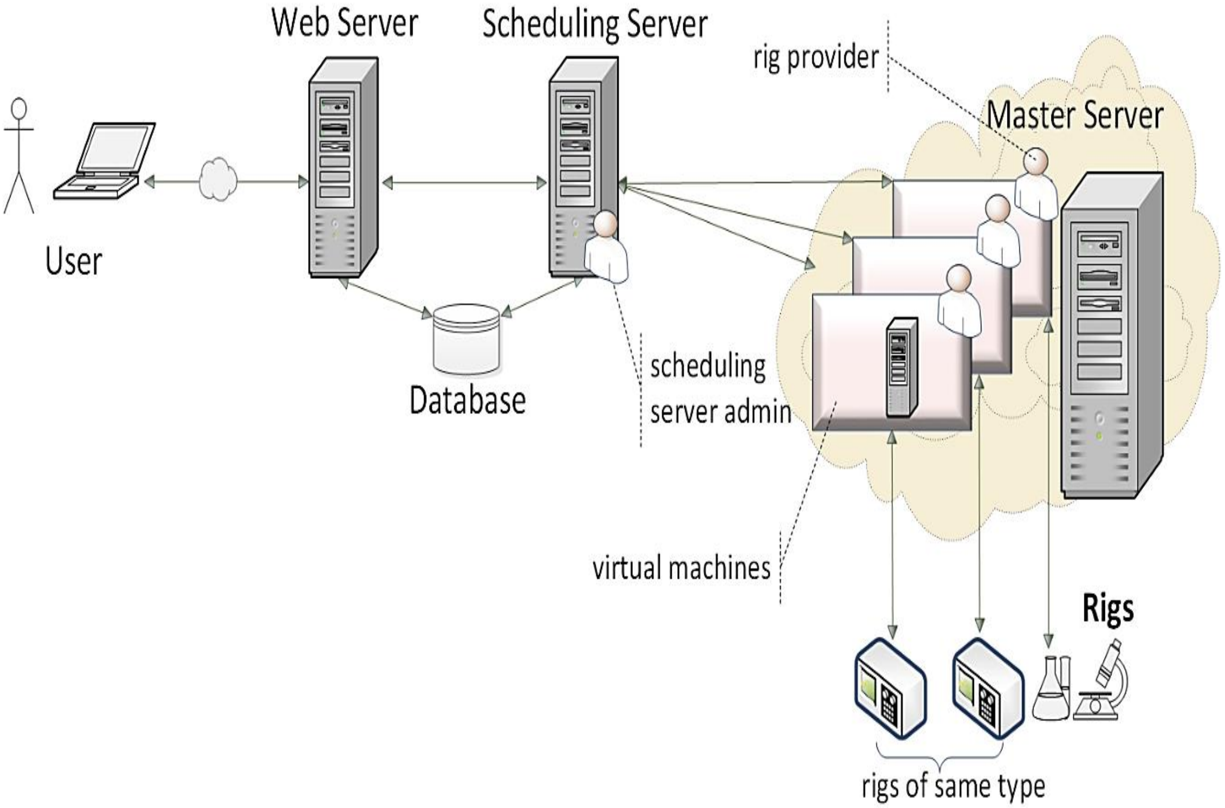


Figure 34. Sahara architecture.

Authentication is based either on a simple local database authentication or through an interface to an institution's local authentication system such as Light-weight Directory Access Protocol (LDAP). Access can be requested and then granted either to an individual rig, or any of a group of identical rigs of a rig type. Each user is associated with one or more user classes. Each user class is assigned the rigs to which they have permissions, which consequently gives their members permissions to queue and/or make a booking for a rig over a period of time. The queue for a rig is based on the priority associated with the user class. If the user has selected to queue for a rig type, they will be assigned the first available rig of that type; if the selection was for a specific rig, they will be assigned only when that rig becomes available. Sahara supports interactive rigs as well as batched rigs, and rigs that use both batch and interactive modes. Additionally Sahara can support monitoring rigs which collect data but do not require user interaction. Sahara allows collaboration on rigs by implementing different user modes. Master users, who are granted access and have full control of rigs, can allow slave users to join a rig session and either allow control (active slave user) or just allow visibility of session (passive slave user). A Webcam is enabled for all user types to monitor experiments.

Sahara and ISA offer similar functionalities but their architectures are significantly different [63]. ISA offers a relative comparative advantage in batched labs and its genius distributed architecture, whereas Sahara offers advantages for interactive labs, queuing, and seamless integration of new experiments. A comparison between the current versions of both architectures is provided in Table 7.

Table 7. A comparison between ISA and Sahara.

	ISA (iLab)	**Sahara (Labshare)**
Developer	MIT	UTS
Web Technologies	Microsoft (ASP.NET, MS SQL, and IIS)	PHP, Java, MySQL, PostgreSQL, and Apache
Compatibility	Only runs on MS Windows Server	Cross-platform
Authentication	Database authentication	Database authentication and LDAP
Client-server communication	TCP/IP	TCP/IP and RDP
Features	➢ user accounts and administrative tools ➢ scheduling ➢ interactive experiments ➢ user tracking ➢ batched experiments ➢ strong support to distributed and federated user account management owing to the functionality of service broker, LSS, USS, and ESS	➢ user accounts and administrative tools ➢ scheduling ➢ interactive experiments ➢ queuing ➢ arbitration of access to multiple identical experiment workbenches
Adding experiments	Complex; Web services API should be design for communication with service broker, LSS, USS, and ESS	Complex; for each experiment GUI should be developed, but simple in case of RDP
No. of universities	3 (Africa), 3 (Australia), 2 (Asia), 5 (Europe), 2 (North America) [60]	5 (Australia) [61]
No. of laboratories added	Unknown	11 in operation and 12 under development [61]

4.5. System-System Communication[1]

As demonstrated in the previous section, each RLMS has different strengths and weaknesses, and potentially addresses a different set of needs. While this diversity could be an asset and should be encouraged rather than avoided, it represents a challenge with regard to supporting cross-

[1] In this section, I would like to acknowledge particularly the contribution of Prof. David Lowe from University of Sydney.

institutional sharing of laboratories between institutions using different RLMSs. Currently, for a remote laboratory that is provided by one institution to be accessed by users from a different institution, typically those users must directly access either the laboratory itself or the providers RLMS (if one exists). This can represent a significant hurdle to effective sharing given that it potentially places a substantial burden on the provider of the laboratory to coordinate access for the users. It can also mean that students who access multiple laboratories (possibly shared from multiple providers) will need to manage numerous access accounts and system interfaces. This can be partially addressed by an architecture that uses federation and single sign-on technologies so that a student from an institution can access the RLMS of other institution through a mechanism defined by the federation and contracts between both institutions. The downside of this is that the students must still access different remote laboratories through different systems. A common portal might address this to some extent, but the system would then potentially lack a consistent look-and-feel.

An alternative is to allow the home institution of the students to maintain their own RLMS, and for this system to be able to interoperate with other RLMSs so that students can directly access provider's experiments. In other words, consider the scenario shown in Figure 35, where the RLMS ISA is installed at MIT and providing access for an MIT user to experiments installed at MIT (MIT 1, MIT 2...).

On the other hand, the RLMS Sahara is installed at UTS, providing access for a UTS user to experiments installed at UTS (UTS 1, UTS 2 …). Adopting a standard Application Programming Interface (API) to communicate between both systems and with any other RLMS, it would be possible for a the UTS user to gain access through Sahara to experiments installed at MIT and integrated in ISA if there is an agreement between both institutions and as long as both architectures support this standard interface in order to interconnect with each other. In response to these needs, a standard Remote Laboratory System Interface Specification (ReLaSIS) that supports system-to-system interaction was proposed by the Global Online Laboratory Consortium (GOLC) [64]. It is designed to be agnostic to RLMSs and to any two RLMSs to interconnect with each other regardless of their underlying technology. Thus, diverse functionality and architectures were taken into consideration. ReLaSIS is based on Web service calls and uses profiles to accommodate different levels of functional support. The common adopted profiles and services calls are provided in Appendix C.

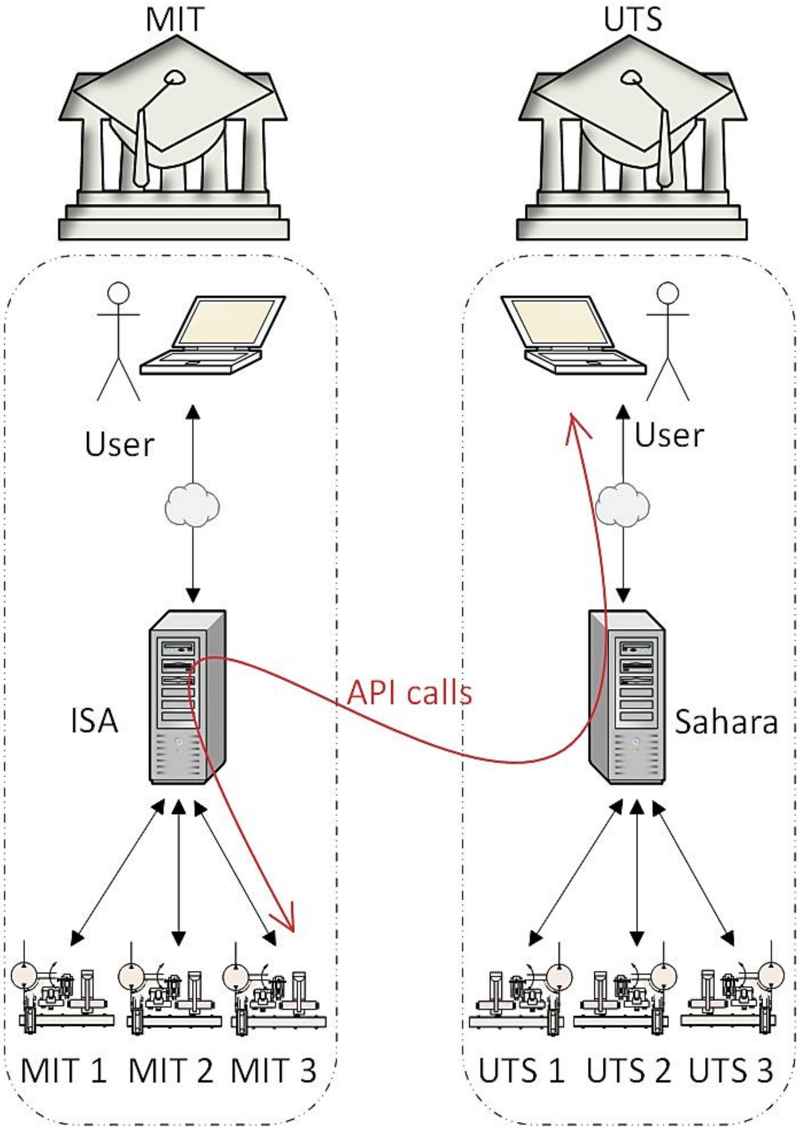


Figure 35. Communication between two RLMSs using API calls.

To illustrate the capabilities that are enabled by these profiles, consider a scenario involving booking and then accessing an interactive experiment as shown in Figure 36.

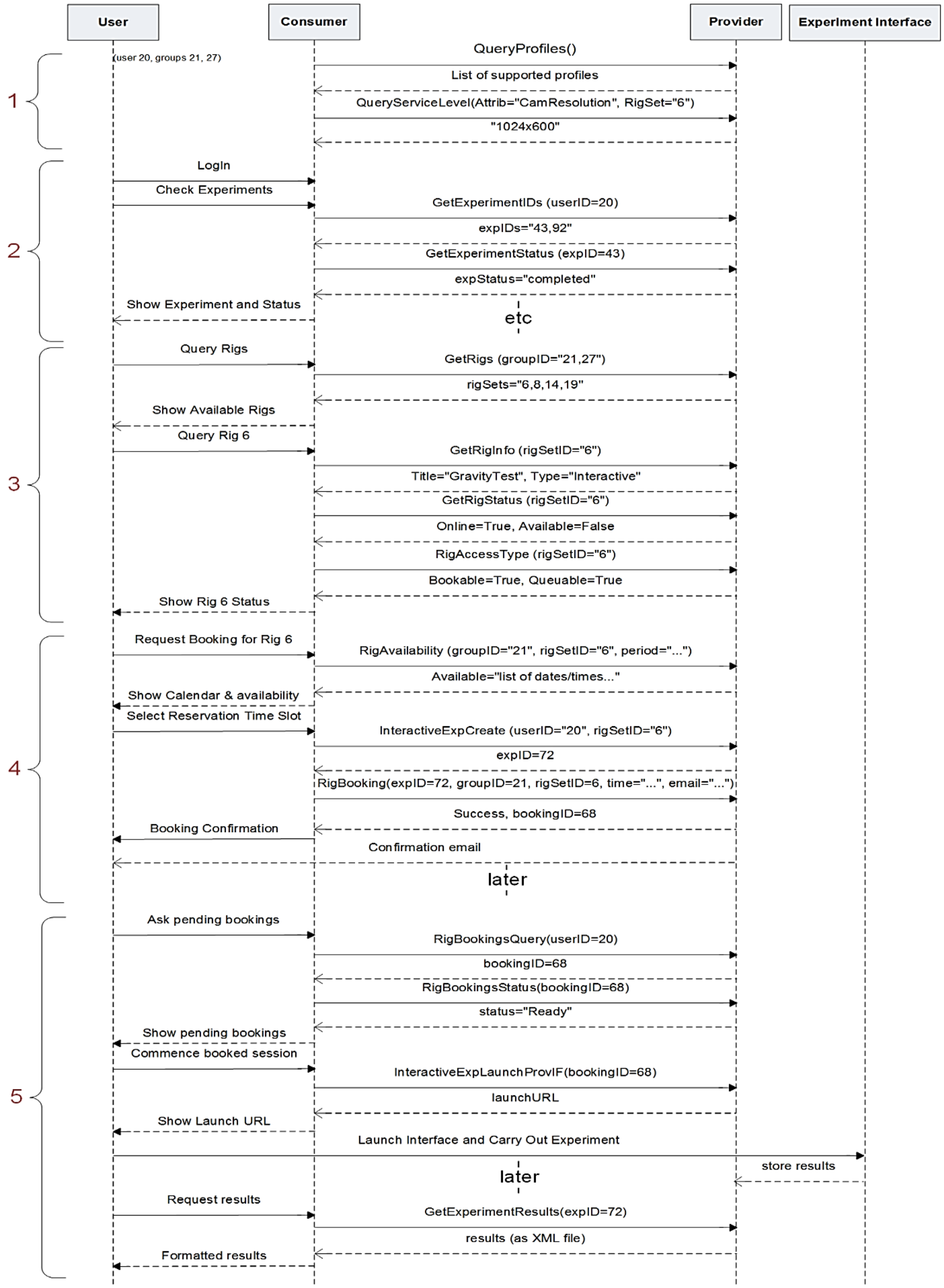

User
Consumer
Provider
Experiment Interface
(user 20, groups 21, 27)
1
QueryProfiles()
List of supported profiles
QueryServiceLevel(Attrib="CamResolution", RigSet="6")
"1024x600"
2
LogIn
Check Experiments
GetExperimentIDs (userID=20)
expIDs="43,92"
GetExperimentStatus (expID=43)
expStatus="completed"
Show Experiment and Status
etc
3
Query Rigs
GetRigs (groupID="21,27")
rigSets="6,8,14,19"
Show Available Rigs
Query Rig 6
GetRigInfo (rigSetID="6")
Title="GravityTest", Type="Interactive"
GetRigStatus (rigSetID="6")
Online=True, Available=False
RigAccessType (rigSetID="6")
Bookable=True, Queuable=True
Show Rig 6 Status
4
Request Booking for Rig 6
RigAvailability (groupID="21", rigSetID="6", period="...")
Available="list of dates/times..."
Show Calendar & availability
Select Reservation Time Slot
InteractiveExpCreate (userID="20", rigSetID="6")
expID=72
RigBooking(expID=72, groupID=21, rigSetID=6, time="...", email="...")
Success, bookingID=68
Booking Confirmation
Confirmation email
later
5
Ask pending bookings
RigBookingsQuery(userID=20)
bookingID=68
RigBookingsStatus(bookingID=68)
status="Ready"
Show pending bookings
Commence booked session
InteractiveExpLaunchProvIF(bookingID=68)
launchURL
Show Launch URL
Launch Interface and Carry Out Experiment
store results
later
Request results
GetExperimentResults(expID=72)
results (as XML file)
Formatted results

Figure 36. ReLaSIS illustrative scenario.

In this example, it is assumed the consumer has contracted with the provider for access to numerous rig sets, but user groups 21 and 27 (to which user 20 belongs) only have access to numbers 6, 8, 14 and 19. In this case, the communication between consumer and provider can be classified into five main stages:

1) In the first stage, the consumer organization retrieves information on the profiles that are supported by the provider and the level of service associated with particular sets rigs, so that this can be used either to provide information to their users, or to adapt the subsequent requests. For example, the camera resolution may be retrieved so that the UI can be adapted, or the policy on information retention might be checked so that the users can be warned when results might be re-moved by the provider.

2) In the second stage, the consumer user number 20 (who belongs to consumer groups 21 and 27) logs into the consumer system. Note that the consumer system is responsible for authenticating their own users and managing the groups that they belong to. Further, whilst the provider gives the consumer system access to certain agreed sets of rigs, it is the consumer systems responsibility to determine which users should be able to access which rigs. The user then enquires about the status of any experiments that he/she previously had created and executed. The consumer system retrieves the list of experiments for this user (43 and 92), and then requests information on each experiment, before presenting this back to the user.

3) In the third stage, the user wishes to carry out a new experiment. They begin by asking to see a list of the rigs to which they have access. They then select a specific rig (ID=6) and check its availability and access type. The response indicates that the rig is bookable and quotable, and whilst it is currently online it is unavailable at that moment.

4) In the fourth stage, the user wishes to make a reservation for Rig 6. The consumer system defaults to showing a booking calendar for the next 48 hours, so queries the provider system for availability over this period. The calendar is shown to the user by the consumer

system, and the user requests a booking at a particular time. The consumer system attempts to make the booking which is then confirmed (bookingID=68). The Provider system also emails the user a confirmation of the booking. Note that the system had to first create an "Experiment" on the provider system that was associated with the booking. This is so that any information required prior to the user commencing a booked session can be incorporated. An example might be configuration or set-up details.

5) In the fifth stage, the user logs back into the system prior to the time of the booked session. They then redeem the booking at the appropriate time and are permitted access to the experiment interface through a URL. Once the experiment is completed the consumer system can request any recorded results from the provider system.

This scenario provides a brief example of how ReLaSIS works. The range of interactions that are supported by ReLaSIS is however much broader than shown in this scenario. Initial or partial implementation results of ReLaSIS is found in [65], where an API was developed for access and manipulate a radio-activity experiment hosted on an ISA server from a Sahara server. The experiment is a batch experiment where user only submits the parameter values—such as radioactivity setup, source name, distance, duration and repeat—and retrieves results later when available. The experiment is registered in the Sahara scheduling server through the API layer and accessed from the Sahara Web interface as an experiment hosted within Sahara—while being hosted at an ISA server—to fulfill the parameter input values. An iLab user will use the API interface as though it is an iLab's lab server. Conversely, a Sahara user will use the API interface as though it is part of the scheduling server interface. However, this particular case was just for demonstration purpose; the communication protocol was relatively restricted, being based on an analysis of the functionality that was common to both iLabs and Sahara, and necessary for this particular laboratory. Nor did the design of the communication protocol take into account the potential for other functionalities supported by other RLMSs. Thus, it did not provide a generalized solution that could support interoperability between arbitrary systems, and with arbitrary apparatus.

4.6. **Discussion**

Obviously, each of the above mentioned scenarios addresses different needs and features services that might not be provided by another [66]. Table 8 provides a general comparison between them.

Table 8. Comparison between different Remote laboratories integration scenarios.

	Integration with metadata repositories	**Integration With LMSs**	**Integration with PLEs**	**Integration with RLMSs**
Indexing (metadata)	✓	✓	✗	✗
Access to laboratories	✗	✓	✓	✓
Standard compatibility	✗	✓	✗	✗
Creating courses	✗	✓	✗	✗
Importing external services and customization	✗	✗	✓	✗
System-system communication	✗	✗	✗	✗

The variety of integration approaches and educational systems is a result of efforts made by several research groups in many directions in order to bring up the best distance education experience. Remarkable results have been achieved but in practice, the definition of today's educational system remains blurred. All the efforts done so far addressing either pedagogical or technical issues have been miscorrelated. As a results, several research directions have been drawn with little impact and far away from institutional agreements and standardizations. E.g., despite the availability of the open source systems such as iLab and Sahara, only few universities managed wrapping all their remote laboratories in one of these systems owing to the absence of an integration pattern to follow and the unclear learning outcomes. The upcoming stream tends to unify all these scattered features in unique consistent educational system—taking into consideration pedagogical and technical concerns—and words such as interoperability and integration has become more pronounced recently.

The parameters listed in Table 8 are the keys for the upcoming features of next generation educational systems. With the wide variety of developed remote laboratories and their increasingly availability in many engineering fields as seen in Chapter 2, indexing them with metadata under defined ontologies has become essential so that they could be easily allocated and discovered.

Importing and exporting services (e.g., learning objects, remote laboratories, etc.) is another important features that future systems should support. Once these services are imported, student could easily mash up them to build his own learning environment either in a formal or informal context. For instance, services could be mashed up to provide scaffolding and other theoretical resources beside remote laboratories sessions (i.e., in the same portal) and hence, more student engagement and better distance education experience.

In a narrow sense, each approach within a particular scenario might have its own exclusive features such as queuing in Sahara, distribution through service brokers in iLab, and semantic Web features in Lab2go. In a abroad sense, integration with open source LMSs is more likely to be the ideal candidate, to build upon, in order to approach a complete educational platform because they typically: allow creating and realizing formal online courses with all the ancillary tools (e.g., administration tools, communication tools, evaluation, and tracking); support eLearning standards such as SCORM; and support metadata standards such as LOM and Dublin Core.

However, unlike PLEs, LMSs are still abide by a monolithic design and lack interoperability with external services or other LMSs. Actually, LMSs are struggling to cope with the demands of today's students who are constantly seeking information in internet from unlimited services and providers. Students cannot be restricted to a single platform, unless it can allow them to import external services and resources.

Currently system-system communication is not completely supported by any educational system despite the contemporary efforts for the following reasons:

- The diversity of system categories (LMS, RLMS, etc.); each system category targets specific features that probably don't exist in the other. For example, LMS supports standard compatibility while no existing RLMS do.

- The diversity of each system within each category. For example, both iLab and Sahara are RLMS but their functionality are totally different; each was designed from a different perspective and outlook and each provides functions or features that are not supported by the other as demonstrated in Table 7.
- The diversity of software technologies adopted in each system an in each remote laboratory. This no more an issue thanks to the variety of middleware technologies. However, this diversity signifies an additional burden to this task.
- Campus access restrictions on their installed systems, either technical or political.
- Each time for a new remote laboratory to be fully aggregated with all its functions, both systems should be modified, which will possibly cause a big burden for the IT staff of both universities.
- If a system server goes down or fail for any reason, all its associated remote laboratories wouldn’t operate and consequently all associated universities that are bridging to this server would be affected.
- Excessive number of communication layers required to tunnel remote laboratories through two systems.

These reasons make system-system communication a very complex solution and not a preferable one, despite of the availability of middleware technologies. An adopted alternative and pathway in this thesis is to focus on lab-system communication instead of system-system communication, considering remote laboratories as loosely-coupled services that are agnostic to systems and clients independently of the provider’s location. All these factors will be considered in defining an efficient paradigm for developing and implementing remote laboratories which will be discussed broadly in the next chapter.

Chapter 5

5. Laboratory as a Service (LaaS)

5.1. Background

So far, the major two strides in the research field of remote laboratories were: (1) extending their application range to encompass sophisticated experiments and many engineering disciplines as demonstrated in Chapter 2; and (2) allowing their integration with educational systems in order to add more services and functionalities as demonstrated in Chapter 4. While these two achievements have managed, to some extent, to mitigate several issues for which remote laboratories were initially chosen to tackle—such as efficient implementation, access anywhere and at any time, safety, and cost reduction—the following challenges have arisen and are yet to be overcome:

- **Interoperability with heterogeneous systems:** current remote laboratory systems aren't able to interact and communicate with other heterogeneous systems—that might be an application container, an educational system (e.g., RLMS, LMS, etc.), or any kind of user client—in a standardized manner and by means of middleware technologies. Fixed and laborious integration of remote laboratories into a specific client or an educational system—without considering further modification in both systems or any further integration possibilities into different systems—has been the prevailing exemplary. Despite the availability of the open source RLMSs such as iLab and Sahara, only few universities managed wrapping all their remote laboratories in one of these systems owing to the absence of an integration pattern to follow and the unclear learning outcomes.

- **Inter-institutional sharing:** sharing laboratory resources among institutions, in order to offset budgets and promote institutional collaboration, is one of the most often raised justification for the use of remote laboratories. Current remote laboratories aren't typically utilized beyond their developer institution. In the literature, all the published articles on remote laboratories report on custom development of authors for specific course purpose. Despite the availability of distributed systems such as iLab, very few institutional sharing of remote laboratory resources have been reported. Even though these systems are open source, their rate of spreading and acceptance is remarkably slow.
- **Coupling with other learning objects and services:** while most of the addressed remote laboratories in the literature have focused on technical and development issues, a little attention have been paid to pedagogical and social issues and even to the learning environment in which remote laboratory should be delivered. The shifting from Web 2.0 to Web 3.0 has changed the way that people interact with the Web and, in our particular case, with online learning environments. Web 3.0 promotes more intelligence, more loosely-coupled services, and more integration of smart things and objects [67]. The current systems however (e.g., RMLS, LMS, etc.) are monolithic with a fixed design and very little flexibility and interoperability. Remote laboratories should adapt to these variations and should be coupled with other learning objects, services or gadgets in a loosely-coupled manner. The fixed design of the current prevailing remote laboratories and educational systems impedes achieving such coupling.
- **Development:** among the principle reasons for which institutions have been reluctant to adopt remote laboratories are that remote laboratories development requires a combination of technical skills (i.e., engineering and IT) which are not regularly found in the curricula of the predominated engineering professors or laboratory technicians. Another reason is that the current solutions developed by universities don't usually rely on well-known standards so that other universities could reuse the design model and build their own remote laboratory leveraging legacy equipment and reusing other available components as long as they adhere to the standards' specifications. Few universities have

benefited from others' developments as in case of VISIR and NetLab, which were discussed in Chapter 2 and 3. VISIR was the most successful remote laboratory project that managed to be implemented at 7 universities from different countries and even continents so far and to establish a community of an increasing number of interested bodies every day. Reasons beyond this success are relying on commercial equipment and software from NI, relying on an open source software that can be easily downloaded and installed, and relying on well-known standards such as VISA and IVI—which gives flexibility in terms of selecting instruments' manufacturer. In other words, the modularized architecture of VISIR allowed partner universities to assemble it in a seamless manner.

- **Standardization:** currently, remote laboratories design and development aren't abide by any type of standard or specification. Adhering to an accepted standardized design pattern would certainly boost their dissemination and facilitate their developing, interoperability, and reusability. This would create also a breeding ground for collaboration among academia and industry.

The above defined issues summarize the main challenges in the development of today's remote laboratories. In response to the above mentioned issues, two novel concepts were developed: (1) modular remote laboratories, based on independent component modules; and (2) Laboratory as a Service (LaaS), a paradigm which aims to deliver modular remote laboratories as a set of loosely-coupled services to be consumed with a high level of abstraction and virtualization—it also defines their broader implementation mechanism. The proposed paradigm would allow overcoming all the above mentioned issues and aims to set foundations for a first global remote laboratories standard model for developers to follow.

5.2. Features

The LaaS paradigm has the following features:

- It divides remote laboratories into component modules and these components all together are delivered in form of a set of loosely-coupled services to be consumed by users.
- It strongly relies on well-known industrial and Web standards and middleware technologies for its entire communications and inter-communications.
- It follows the Service Oriented Architecture (SOA) in terms of defining the relation between laboratory providers, laboratory consumers, and service broker or repository in which remote laboratories are registered and indexed under metadata standards and ontologies.
- It merges features from cloud computing—in terms of consuming services on demand with minimum restrictions and higher virtualization—and features from grid computing—in terms of global distribution.
- It embraces the Web of Things (WoT) in terms of coupling with heterogeneous services and bringing objects to the Web for all spectrum of needs—in either formal or informal contexts.

5.3. Objectives

In clearer terms, the purpose and the overall goals of the proposed LaaS paradigm can be summarized in the following points:

- Define an organized manner for sharing remote laboratories globally among institutions.
- Allow wrapping remote laboratories in any heterogeneous application container (e.g., widget, applet, or any Web client) independently of the underlying technology adopted in both, as well as, their coupling and mashing up with heterogeneous services (e.g., learning objects) across the Web.
- Facilitate maintenance, reusability, and leveraging legacy equipment.
- Allow interchangeability of components between provider and consumer—seamlessly and programmatically— insofar as consumer could contribute with one or more component instead of the fully-reliance on the provider's equipment and facilities.

- Promote online experimentation and discovery in either every day's formal or informal contexts.
- Set principles for a first global standardized design pattern—for remote laboratories development and implementation—to be followed and adopted by remote laboratories developers.

5.4. Related Works

Earlier attempts to deliver remote laboratories as a service can be found in [68-72]. In [68], the functions of commercial instruments based on VISA and IVI were listed in Web Services Description Language (WSDL) files and registered in a Universal Description, Discovery and Integration (UDDI) to be allocated and consumed by Simple Object Access Protocol (SOAP) Web service. A similar approach for controlling instruments online using SOAP Web service is found in [69, 70]. In [71], a simple experiment was developed in LabVIEW and delivered as Representational State Transfer (REST) Web service, to be consumed using Asynchronous JavaScript and XML (AJAX) calls. A similar approach based on REST Web service and AJAX is found in [72].

A distinctive approach was adopted in [73], where a further effort have been realized in order to add intelligence to remote laboratories at the server side and to make little or no assumption about the client. The underlying communications in this approach were realized using Websockets owing to its efficiency and high transmission rate. On the other hand, to promote compatibility with different client applications.

It is also worth noting that the acronym LaaS has been pronounced in the literature for almost three years (i.e., concurrently with the elaboration of this thesis) few times, with two different interpretations. The first interpretation refers to the cloud computing and the Anything as a Service (XaaS) concepts as described in [74-78]. In these approaches, however, the difference between cloud computing and remote access is still blurred. Yet, there is no clear application of the cloud computing principles on real physical laboratories that might be distributed at various universities globally. The second interpretation—the interpretation assumed in this thesis—simply refers to the delivery of remote laboratory as a service that can interoperate with heterogeneous systems and services. The second interpretation is more generic and can be implemented many ways. For instance, in [79], Web service was adopted in conjunction with a proprietary Lab Description Language (LDL) developed by the author in order to achieve interoperability.

Even though, the solution proposed in this thesis (LaaS) builds on top of these efforts, it has four main distinctive aspects. The first aspect is that Web service in LaaS is a method and not a solution itself and its adoption is not necessary. For instance, for data streaming (e.g., video and measurement streaming) low level protocol communications are implemented instead. The second and most important aspect is that LaaS goes further beyond abstracting the functions of the laboratories; it implies their development as a set of independent component modules in order to allow interchangeability of components between providers and consumers—seamlessly and programmatically—insofar as consumer could contribute with one or more component instead of the fully-reliance on the provider's equipment and facilities. The third aspect is that LaaS contemplates the future Web and the next generation learning environment in terms of: (1) seamlessly allocating and importing services; (2) bringing objects to the Web; and (3) mashing up and coupling services together—which was possible, in part, thanks to the modular remote laboratories concept. The fourth and last aspect is that LaaS is meant to be a model that addresses the development of remote laboratories, as well as, their implementation process broadly—which entails the relation between consumers, providers, and service broker, as well as, the format of exchanging information and resources between them. As a Proof-of-Concept, a modular remote laboratory was developed successfully and implemented according to the LaaS paradigm.

5.5. Context

A first task in defining a paradigm for developing and implementing modern remote laboratories is to study thoroughly and to understand the nature of the more likely to be: the Web of tomorrow; the next generation learning environment [80] in which remote laboratories will be delivered; and the possible methods of access. This study will be realized in the following three subsections.

5.5.1. **ELearning 3.0**

Web technologies are the core part of eLearning technologies and both are continuously progressing alongside, e.g., the current Web 2.0, also known as social Web, spur the emergence of eLearning 2.0 and enables a high level communication and collaboration among students online. While the read-only Web, Web 1.0, has revolutionized the way people access to information, the second generation of the Web or Web 2.0 has revolutionized the way people communicate with each other across Internet. Web 2.0 allowed users to participate in creating and publishing content and communicate, collaborate, and share information with each other via blogs, wikis, and social networking sites. Web 2.0 has transformed the Web into an interactive environment that provides richer user experiences and allowed the combination of disparate information in a variety of data formats. Features of Web 2.0 can be classified as follows:

- **Interactivity:** dynamic interfaces using JavaScript and enables interactive features within Web pages in form of widgets, gadgets, and snippets by means of plugins. As a result of this progress, the gap between Web and desktop applications has been diminishing.
- **Communication:** asynchronous communication tools, such as wikis and blogs, that don't require users to be simultaneously connected at any given time and synchronous communication tools such as chat.
- **Contribution:** Sharing content in multimedia format across large-scale Web sites such as video posting Web sites and photo-sharing Web sites.
- **Mashup:** piping Web resources in heterogeneous formats and from scattered sources together to create a novel and rich application.

Up to date, Web 3.0 is still purely hypothetical and intends to capture the trends and characteristics beyond Web 2.0. It is anticipated that the driving force for Web 3.0 will be linking contents' data to making online information machine-readable and facilitating Web accessing by anything, anywhere, and anytime using a variety of devices and objects. The three main speculated streams that envision the Web of tomorrow are:

- **Semantic Web:** Web 3.0 will be a system with artificial intelligence that understands the meaning of its contents in order to aid users finding information more easily. Web 3.0 will not only be human-interpretable but also machine-interpretable. Search engines will no more be restricted to syntax-based search queries but they will be able to grasp the semantics of the information on the Web. Contents and services on the Web will be defined using sets of rules expressing logical relationships between concepts and data (ontologies) which enables human knowledge to be machine-readable. Ontologies form a formal and explicit representation of concepts of a certain domain and the relations between these concepts that can be distinguished in a particular situation. This explicit and formal specification allows machines to understand and interpret the information. Web ontology languages that allow users to add structured metadata to the Web are RDF and OWL, both are based on XML.
- **Pervasive or ubiquitous Web:** The number of smart objects, providing context-awareness of the environmental conditions that surround us, increase every day. These objects sense real-time information and collect and process data that are used later on for providing information and taking decisions. This is possible due to their embedded sensors, networking, and computing capacities. The increased computational power of mobile devices, the ubiquity of wireless networks, and the development of modern wireless sensor technologies leaded to the exciting vision of the interconnected smart objects or to a promising Internet of Things (IoT). The introduction of IPv6 facilitated the effective penetration of the Internet in embedded computing allowing internet applications to identify and communicate with all existing devices because of the extremely large address space of the IPv6 protocol. Examples of smart devices and objects are wireless sensor networks, ambient devices, household appliances, and Radio-Frequency Identification (RFID) tagged objects. A refinement to IoT was WoT, which doesn’t only consider integrating smart things into the Internet, but also it consider integrating them into Web applications as first-class citizens of the Web. WoT is about reusing the Web standards to connect the quickly expanding ecosystem of embedded

devices built into everyday smart things surrounding us. Smart things are connected to the Internet by embedding Web servers directly onboard in order to expose their functionalities as Web resources by applying Web middleware technologies such as Web services. This allows creating loosely coupled services so that they can be easily reused over the Web. In a number of cases (e.g., Bluetooth, Zigbee, or RFID tagged objects), it makes sense to expose services of devices with limited capabilities as Web services through an intermediary component (e.g., proxies or smart gateways) that understands their device specific protocols. A smart gateway is actually a Web server that abstracts behind Web services API the actual communication between devices and the gateway. In short, internet will move beyond being a network that connects computers together to a network that connects everything surrounding us together. Physical things will be fully Web-enabled thanks to the expanding ecosystem of embedded devices built into everyday smart things that are integrated into everything in our live, e.g., clothing, appliances, and automobiles. Users will be able to access information and interact with every connected device in a ubiquitous way. Web will be aware of the environment and targeted physical things and will respond proactively. Information will be available anytime, anywhere, and anyhow—not only common desktops but also all types of devices that can someway be Web-enabled.

- **Mashup:** while mashup has already been a recent feature of Web 2.0, it will be strongly embraced in the future Web. The future web is expected to be composed of a mesh of interoperable and semantic Web services that can represent everything in our life ranging from software applications to objects and even humans. These will not only be integrated together but they should be able to communicate with each other. mashup is the combination of data, presentation, or functionality from multiple sources (i.e., internal and/or external) to create new applications or services. The main characteristics of mashup are combination, visualization, and aggregation. mashup composition is usually simple enough to be used by end-users and generally doesn't not require programming skills and rather support visual aggregation of GUI widgets, services, and components together within a customized Web page, as in iGoogle (www.google.com/ig). Else, mashup composition can be data-centric [81], where similar types of data is extracted from multiple sources and transmitted to the user's browser in its final form as a single representation, as in HealthMap (healthmap.org). Services could be combined and integrated at client-side— where external sources are invoked and directly embedded into the client—or at server-side—where internal server functions as a proxy between the client and respective providers and it is responsible for buffering and cashing data. The server-side model enhances security and is highly relevant for enterprise applications. By using semantic annotations, mashup is transformed into semantic mashup, where services are defined using ontologies in order to improve the possibilities to allocate and match between them.

As a result, the future Internet will embrace the artificial intelligence, the omnipresence of everyday connected objects, and the loosely coupled services. Combining all together will form a bridge between the virtual and the real world. It is very likely that these three speculated mainstreams will co-exist in the near future and will form the next generation of the Web. This could give us insights about the characteristics of online learning environments of eLearning 3.0.

5.5.2. Service-Oriented LMS

The shift from Web 2.0 to Web 3.0, and consequently from eLearning 2.0 to eLearning 3.0, has demanded a major rethinking in the current online learning systems, commonly LMSs. LMSs initially released as black box solutions, in a proprietary format, for a specific purpose. During the time, standards such as Dublin Core, IEEE LOM, and IMS LRM for interoperability between different LMSs at content level have emerged. Later, the demand for modularized and personalizable e-learning platforms has grown, which yielded the release of open source solutions and customization toolkits. The focus wasn’t only on sharing content but also on sharing learning objects and courses, which yielded, on the meantime, the release of new standards such as SCORM and IMS Content Packaging [82].

Nowadays, there exist several factors that are shifting the path of development of LMSs, among them [82]: (1) today's LMSs are failing to keep pace with advances in Web 2.0 technologies and social interactions online; (2) today's LMSs can't support architectural flexibility due to their monolithic designs, information are only exportable and importable across different LMSs but not interchangeable among heterogeneous systems dynamically; (3) lack of interaction with learning environments that are living outside the LMS ecosystem such as mobile devices; and (4) lack of integration with remote laboratories.

These factors have led to the learner-centered approach, PLE, in which users apply Web 2.0 services to create their own learning management environment in instead of passively consume imposed learning materials. PLE are provides the learner with the freedom of personalizing and managing his/her own learning environment at his/her own pace and without institutional restrictions converting him/her from only a passive learning consumer to an active learning consumer and provider. PLE is based on a set of loosely coupled learning resources and components that are customized with regarding to the learner's necessities. PLE approach could be realized either on a separate platform or on a customizable LMS—for example, integrating Google Wave Gadgets into Moodle.

Service-oriented LMSs extend the capability of the ordinary LMSs and embrace the PLE approach in order to support a wider range of interoperability between architectures and between online services. They support both formal and informal customizable learning environments. Systems will no long share contents and learning scenarios solely but also will exchange tools, functionalities, semantics, and control seamlessly and dynamically as shown in Figure 37. Next generation educational systems emphasize the use and adoption of external services for both formal and informal learning.

In order to support technological diversity, several initiatives such as E-Learning Framework (ELF) (elframework.org), Open Knowledge Initiative (OKI), IMS Abstract Framework (IMS AF), and IMS Tool Interoperability (IMS TI) have defined initial steps toward service-oriented e-learning platforms.

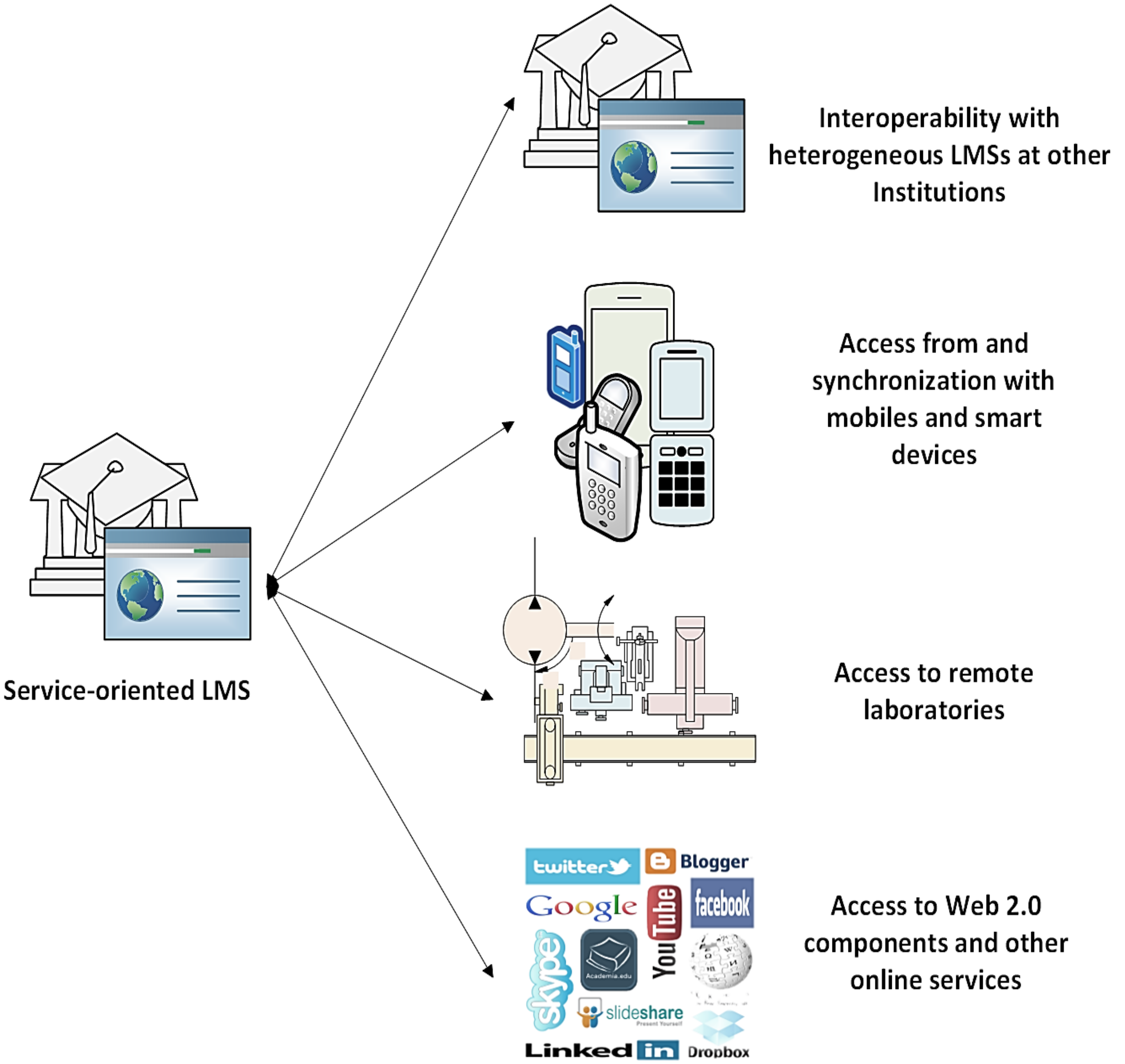

Service-oriented LMS
Interoperability with heterogeneous LMSs at other Institutions
Access from and synchronization with mobiles and smart devices
Access to remote laboratories
Access to Web 2.0 components and other online services
twitter
Blogger
Google
YouTube
facebook
skype
Academia.edu
slideshare
LinkedIn
Dropbox

Figure 37. Service-oriented LMS

ELF illustrates e-learning systems' common functionalities as services in order to support SOA. The OKI project creates a standard architecture of common services that learning software systems need to share, such as authentication, authorization, and logging. A suite of interfaces specifications known as Open Service Interface Definitions (OSIDs) has been developed and published with the purpose of providing interoperability among systems with different underlying technologies. OSIDs are software contracts based on SOA. They reduce the integration cost, promote the software reusing, ease the software development, and allow institution to be concerned with the core design of the system rather than being burdened with the common functions. OKI describes a complete set of services between the LMS (service consumer) and the application (service provider).

The IMS AF is set of specifications to build a generic e-learning framework that is able to interoperate with other systems following the IMS AF specifications. IMS AF describes the e-learning system as the set of services that need to be offered. IMS AF is a standard that can be complemented by the OKI OSIDs because OKI provides more specific information about the semantics of the services, how to use them and in what kind of situations they could be used.

While ELFs, IMS AF, and OKI work on the exchange of information and services, IMS TI focuses on the process on how a remote service is installed on a web based learning system. The IMS TI specification resolves the way an external application can be integrated inside a LMS. IMS TI describes the specific Web Services calls necessary to conduct between service provider and consumer, so the consumer (i.e., LMS) can deliver the providers application to its user in a consistent way. In other word, it focuses on how the teacher and the student reach the application form the LMS in a way that the LMS users students and, especially teachers, could perceive the external application like a service native to their LMS [83, 84].

5.5.3. Mobile Access

The last study prior delving into the description of the LaaS paradigm is on the possible methods of access. In addition to the traditional computer-access method, mobile access is no longer a minor matter. This is obvious owing to the release and the high increasing proliferation of smart phones and tablets in since only few years ago. It is, in fact, foreseen to be the primary access method within the next few years. The most wide-spread mobile operating systems are iPhone OS or iOS, Android, Windows Phone, Symbian, BlackBerry OS, and Bada. The Mobile applications or apps are classified as native app, Web app, and hybrid app [85].

Native app are binary executable files that are downloaded from app stores and installed on the device. Native apps have access to all the APIs that the device's OS allows and uses all the facilities of the phone including camera, Global Positioning System (GPS), accelerometer, magnetometer, etc. Native apps are developed using Software Development Kit (SDK) provided by the OS vendor. Each SDK accepts a specific programming language or more (e.g., C++ for Bada, Windows Phone SDK, Symbian SDK, or BlackBerry SDK, Java for Android SDK, and Objective-C for iOS SDK).

Web app are Web-based applications developed using Web standards such as HTML, Cascading Style Sheets (CSS), and JavaScript. Most mobile vendors utilize the open source Web browser engine in their browsers WebKit that provides the most comprehensive implementation of HTML5, which results in powerful and rich browser-based apps and compatibility across different devices and Oss as well. Web apps may be used offline and online and can give access to some device’s sensors information data, handle screen touch, play media files, and communicate with other pages. With canvas element it is possible to rendering of 2D shapes and bitmap images dynamically with good performance and with WebGL (www.khronos.org/webgl/) it is possible to rend interactive 3D and 2D graphics. Web apps can be launched from a shortcut that is indistinguishable from that used to launch native apps. A growing number of JavaScript toolkits have been created that generate user interfaces that are comparable in appearance to native app such as jQuery Mobile (jquerymobile.com/).

Hybrid app merge native approach with Web approach by wrapping a Web-based portion inside a thin native container that provides access to device features. Hybrid apps allow user to maintain direct access to native APIs while using cross-platform Web apps. They use OS APIs to create an embedded HTML rendering engine that serves as a bridge with the Web browser. App developers can choose between coding their own bridge or taking advantage of ready-made solutions such as the open source library PhoneGap (www.phonegap.com) that provides a uniform JavaScript interface to generate apps for every supported platform.

Native apps are faster than interpreted languages such as JavaScript (i.e., because they are compiled) and have full access to the device features. However, their critical disadvantage is that they are confined to a certain OS and for developing a specific app for a specific OS, certain languages and SDK should be used, making their development and maintenance a very long and expensive undertaking. The problem of hybrid apps is that neither they provide the performance of native apps nor the compatibility of Web apps. Web apps are multiplatform support, don’t require the use of any SDK— even though there are no standard tools to assist the developers, and have low development cost. Web apps runs within the browser—which is a native app itself that has direct access to OS APIs but only limited number of these APIs are exposed to the web apps—and have partial access to the device’s features. Although, with the advancement in HTML 5, Web apps are expected to be pure HTML5-based and to have access to all device’s features. In general terms, in the future it is expected that Web browser will be the primary software environment for most purposes either on desktop computing or on mobile devices. Therefore, modern remote laboratories should be accessed from Web apps and their interface should be purely built upon Web native technologies in order to support the majority of mobile devices.

The previous three studies could give us insights about the development and implementation context. Accordingly a paradigm for developing and implementing modern remote laboratories suited for this context can be defined as demonstrated in the next sections.

5.6. LaaS Description

Next generation online learning environments are more likely to be based on “semantic and ubiquitous loosely-coupled Web services of everything”, meaning that Web services—as learning objects—will be interoperable, will be easily discovered, and will represent any kind of virtual or physical object. Among an unlimited pool of services in the World Wide Web (WWW), students should be easily able to discover and select the services that most suit them and aggregate them together to build their learning environment in either formal or informal context and these services should be able to interact and communicate with each other in order to build more powerful applications. For instance, consider the scenario depicted in Figure 38 in which student aggregates a laboratory widget from a provider and an Excel sheet widget from another provider.

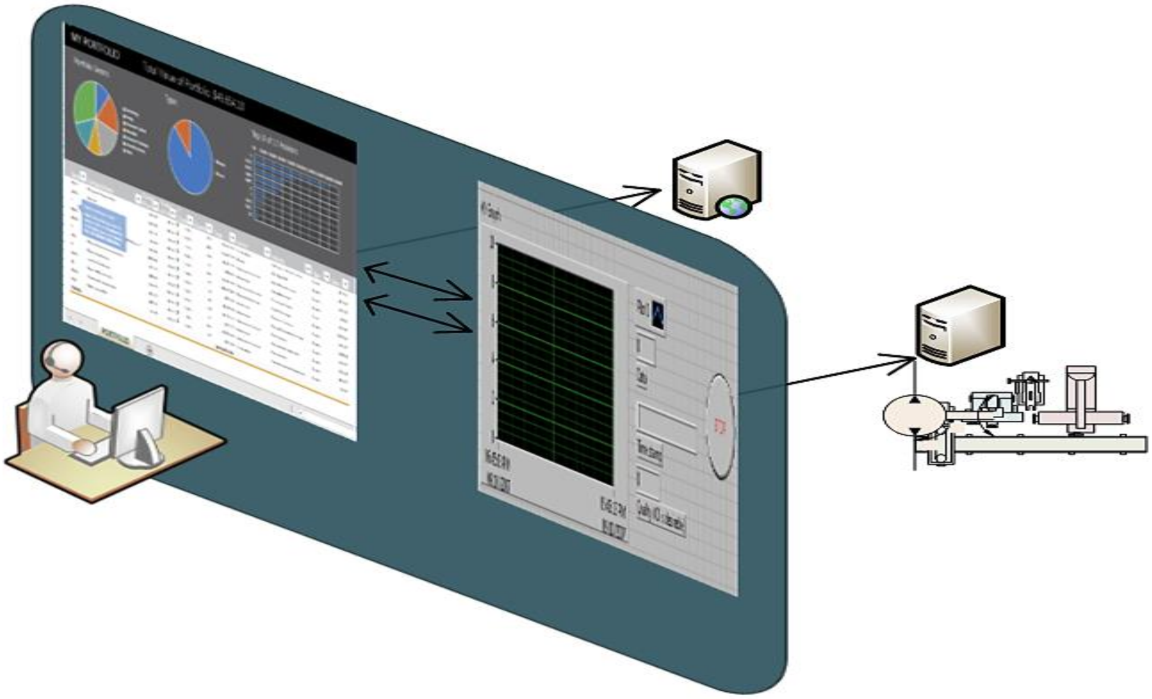


Figure 38. Mashing up imported services.

If both widgets provides an abstracted set of services for their functions, it would be possible to mesh both services in order to build an enriched application where laboratory results are analyzed in the excel sheet. Thus, a remote laboratory should be considered as a service that could be complemented by other services such as Chat, videos, calendar, etc. These scattered services (i.e., remote laboratories in our particular case) should possess the following characteristics: easily discovered and loosely-coupled. The term “loosely-coupled” encompasses the interoperability with heterogeneous learning environments and the interoperability with other services within the learning environments as well. In other words, remote laboratories should provide APIs for clients and for other services depending on the functionality of each remote laboratory. For instance, consumer might consume part of the provided services to develop widget as GUI and the other part for developing another widget for drawing graphs from the received measurement.

Another important characteristic is the "easily discovery", which can be achieved by adding semantics to the exposed APIs of the remote laboratories—known as semantic Web services. However, this would add extra layers to remote laboratories APIs and would increase complexity and thus, it is not the preferable solution in the LaaS approach. A better alternative is to rely on a metadata repository with semantic Web features as described in Lab2go, which would allow discovering remote laboratories and exporting them as well. This would facilitate to providers adding ontologies to their services easily via Web instead of programming. This would also create a vast cloud of scattered remote laboratories from different providers as services.

In cloud computing service provider delivers its virtualized (i.e., on virtual machines) pool of resources or services to customers on demand, everywhere, and at any time abstracting to them the physical layer and technical concerns. Virtualization facilitates dynamic load balancing considering the potential for automatic spawning and closing of virtual machine instances in regard to application load and performance requirements. Customer can access to services on demand without need for interaction with cloud service provider. Resources are elastically scaled upon customer request without any limits and customers are charged accordingly. Cloud resources are measured, audited, and reported to the customer based on a metered system. Cloud computing reduces IT administration burdens and costs and enables application development and implementation in an efficient way. Cloud deployment can be private (i.e., either on- or off-premises), public, or hybrid. The three cloud service models that have been universally accepted:

- **Infrastructure as a Service (IaaS):** provides users with administrative tools and Web-based access to fundamental computing resources such as servers (i.e., as virtual machines), disk images, and networks. Examples of IaaS service providers include: Amazon Elastic

Compute Cloud (EC2), Google Compute Engine (GCE), Windows Azur virtual machine, and Oracle Infrastructure as a Service (Oracle IaaS).

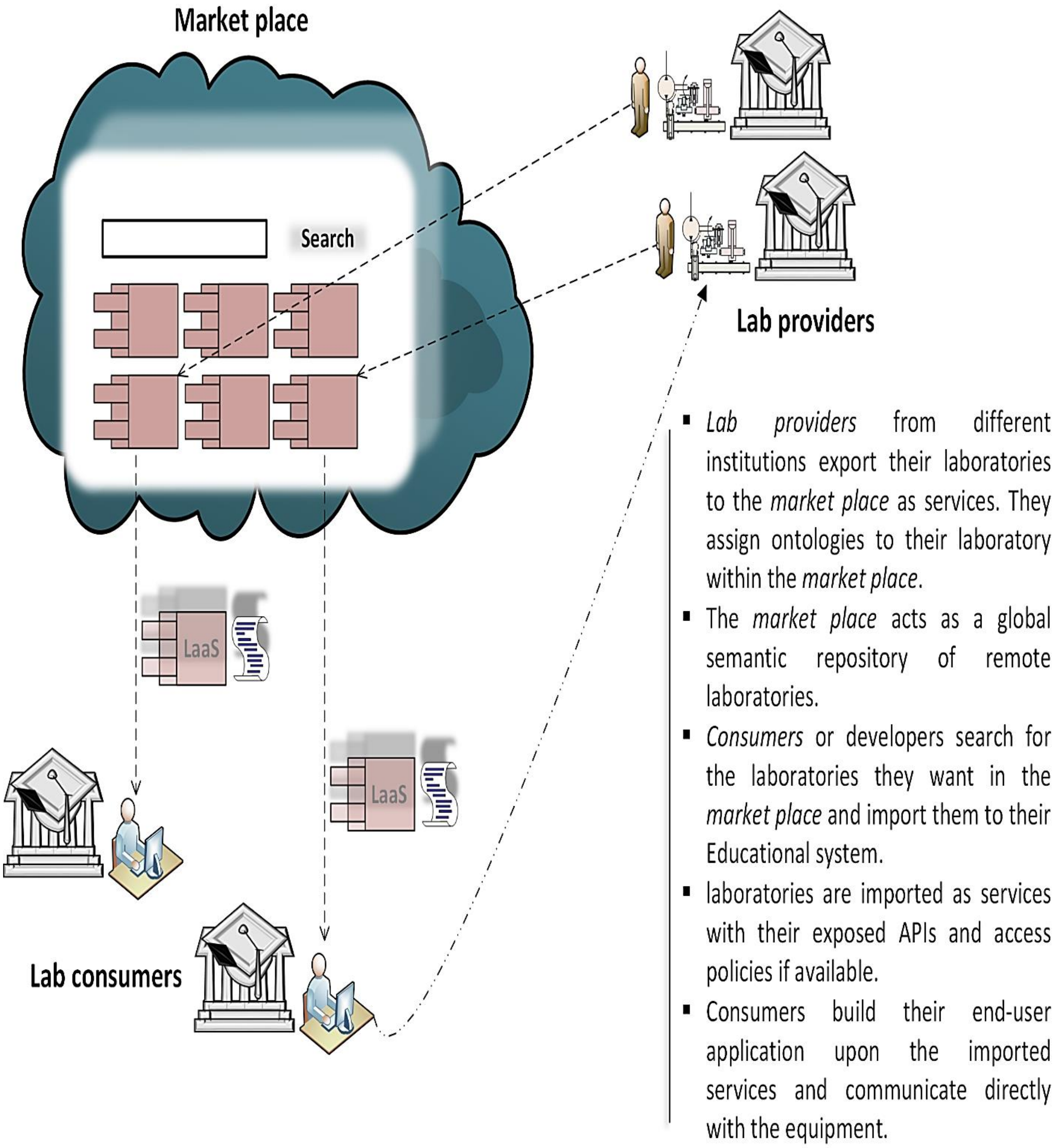
- **Platform as a Service (PaaS):** provides users with administrative tools and Web-based access to a virtual machine with an operating systems (i.e., including a Web server and a database), where he/she can install and deploy his/her applications. Examples of IaaS service providers include: Windows Azure cloud services, and Google App Engine.
- **Software as a Service (SaaS):** provides users with administrative tools and Web-based access to a Web application software. Examples of IaaS service providers include: Google Apps, Microsoft Office 365, GMail, Google Docs, and Dropbox.

From the cloud computing perspective, LaaSs should be registered at an intermediary and provided to users on demand abstracting to him the physical layer and the technical concerns. However, LaaSs would be hosted physically at their distributed providers but allocated and discovered at the broker. Thus, LaaS paradigm is partially cloud and partially distributed and servers of each laboratory will still be located at their provider’s institution. The cloud will be a global semantic repository server denominated “market place” as depicted in Figure 39. The market place is a repository that at least creates providers or institutions profiles and supports semantic Web technologies—or at least provides an enhanced search-engine.

Applying LaaS paradigm will help tackling many concurrent remote laboratories issues. Remote laboratories will be easily indexed and discovered in a global repository similar to YouTube for videos, Flickr for photos, and Wikipedia for information. Each institution would be able to create its own account and add their laboratories in form of a set of services along with their access policy.

If the laboratory is made available, it should be accessible unless another session is currently running by another user. Else, the consumer should contact the provider for enquiries. It is left up to the consumer to build his/her own scheduling system for a large scale deployment with numerous groups and students. Scheduling system is out of the scope of the LaaS paradigm as it focuses on the lowest level implementation for maximum abstraction. Once the consumer has imported the

LaaS he/she wants, he/she would be able to consume its APIs and communicate directly with the equipment without any extra layers in between.



- *Lab providers* from different institutions export their laboratories to the *market place* as services. They assign ontologies to their laboratory within the *market place*.
- The *market place* acts as a global semantic repository of remote laboratories.
- *Consumers* or developers search for the laboratories they want in the *market place* and import them to their Educational system.
- laboratories are imported as services with their exposed APIs and access policies if available.
- Consumers build their end-user application upon the imported services and communicate directly with the equipment.

Figure 39. Laboratory as a Service (LaaS) model.

5.6.1. Implementation

In the LaaS paradigm, the relationship between consumers, providers, and the market place or the service broker follow the SOA. SOA is a software architecture design pattern for developing applications by combining loosely coupled and interoperable services provided by distributed computers connected over a network—instead of implementing the whole solution from scratch. SOA separates the users from service implementations, by using interface definitions that hide the implementation of the language-specific services. As a result, SOA-based systems are not tied to a specific technology and services can interoperate across diverse hardware platforms, operating systems, and programming languages and be accessed across networks. For example, services written in C# running on .NET could be consumed by an application written in Java and running on Java EE platform and vice versa. SOA enables reduction of time and cost in software development and reduction of application and infrastructure complexity as well. It aims to facilitate software reusability, promote standardization business between enterprises, increase profits, improve product quality, increase customer satisfaction, and achieve operation agility. SOA defines a relationship between the service provider, broker, and user. The service provider creates a service and publishes its interface and information to the service registry (or the service broker). The service broker then expose and lists all the potential services available within it. It could be either public or private that are only accessible to a limited audience, e.g., users of a company intranet. The service consumer (or the client) locates entries in the broker registry then binds to the service provider in order to invoke its service/s. The service consumer can access multiple services (i.e., and from various providers).

Thus, the essential SOA requirements are: (1) interoperability of services regardless of their platform, operating system, and programming language; (2) description of services, their characteristics, and the data that they exchange in a clear and unambiguous manner that allows a potential consumer to allocate and consume them; and (3) access to services by means of standard communication protocols and common format messages. The most popular middleware technologies for implementing higher level SOA are Common Object Request Broker Architecture

(CORBA), Distributed Component Object Model (DCOM), JAVA Remote Method Invocation (RMI), Data Distribution Service (DDS), and Web services. Software specifications—targeting high level business applications—like Java Business Integration (JBI), Windows Communication Foundation (WCF), and Service Component Architecture (SCA) provide a model for composing applications that follow SOA and thus facilitate developing SOA solutions.

> *i.e., SOA has become relevant also for smart home architectures, automation, industrial sectors, and IoT in which devices, sensors, and actuators are connected to network by discovery protocols such as Universal Plug and Play (UPnP), Bluetooth, RFID, ZigBee, Jini, and Devices Profile for Web Services (DPWS). Modularization frameworks for these purposes that target modularity at lower level include Open Service Gateway initiative (OSGi) and Microsoft Managed Extensibility Framework (MEF). These frameworks provides the ability of building applications from a number of modular, reusable and collaborative components that can co-exist in different versions, communicate with each other over a service bus, and can be dynamically reloaded (as plug-ins).*

CORBA is an object-based distributed middleware specification, established by the Object Management Group (OMG) (www.omg.org), which allows client to invoke operations on distributed objects regardless of the type of their programming language, architecture, or operating system. It consists of a client, a server and an Object Request Broker (ORB) at each side. The client's ORB is responsible for locating the target registered CORBA objects and matching the requesting client to their ORB and server. Once established the communication, the client's ORB marshals the arguments and routes the invocation out over the network to the remote object's ORB. The remote object's ORB then invokes the method locally on the server and sends the results back to the client via the network. ORB uses Internet InterORB Protocol (IIOP) to communicate and Interfaces Definition Language (IDL) to describe an interface that is independent of programming language. The IDL code is compiled into an OS- and language-specific generated code classes for use by the user application. CORBA not only provides a mechanism to define interfaces between components, it also specifies standard services such as transactions and security, events, time, and other domain-specific interface models which describes the interoperability feature for CORBA compliant applications. Most object-oriented languages in use today have an API that allows CORBA to be used. DDS is another middleware technology defined by OMG and provides a service more suitable for high-performance real-time asynchronous and dynamic operations. The standard is being increasingly used in a wide range of industries and big data applications. It is more suitable for data dissemination to many nodes in dynamic environments and provides a very rich set of Quality of Service (QoS) that allow to control every aspect relating to data. It is data-centric, whereas CORBA is object-based.

DCOM is the distributed version of Microsoft's Component Object Model (COM) technology which allows the creation and use of binary objects/components from languages other than the one they were originally written in. DCOM it a proprietary Microsoft technology currently supports Visual J++, C++, Visual Basic, JScript, and VBScript. It is layered on top of the Distributed Computing Environment (DCE) Remote Procedure Calls (RPC) specification. Unlike COBRA which tackles at the source code level, DCOM addresses interoperability among binary software components. DCOM provides a secure, reliable, and high performance environment. Similar to CORBA, IDL is adopted for defining COM’s interfaces and classes. .Net Remoting superseded DCOM as DCOM was designed before the Internet became a priority for distributed systems .Net Remoting eliminates the difficulties of DCOM by supporting different transport protocol formats and communication protocols.

Java RMI is a Java API that allows sharing of Java objects between Java Virtual Machines (JVM) across a network. A remote stub acts as a local representative or a proxy of the remote object in the client’s JVM. The stub implements the same set of remote interfaces as the remote object itself. When the methods of the stub are invoked by the client, it carries out the calls processing on its remote object. This allows a stub to be cast to any of the interfaces that the remote object implements. Unlike CORBA, the RMI does not require infrastructure for its implementation beyond the JVM, which is equivalent to the CORBA’s ORB. OSGI is one of the implementation of Java middleware in smart homes. The core advantage of Java RMI include its support for interoperability in terms of interoperation execution and information exchange, and full support for modification of existing systems. As a feature, RMI not only allows remote method calls of Java objects, but also remote method calls of CORBA objects in order to achieve interoperability between RMI and CORBA applications. The underlying protocol of Java RMI is Java Remote Method Protocol (JRMP).

DCOM relies on a proprietary binary protocol, which isn't supported by all object models and thus hinders interoperability across platforms. It has been defined and implemented as a standard only on Microsoft's Windows platform and it is limited to Microsoft's .NET programming languages. This tight integration made development easier but at the same time extending applications beyond Microsoft's Windows platform is essentially impossible. Moreover, DCOM usually requires a highly administered, costly, and complex environment to implement and manage. Similarly, Java Middleware can only be implemented with the presence and requirement of JVM in remote and local components of the system involved. CORBA is relatively advantageous as it allows developers to choose almost any language, hardware platform, and networking protocol. However, its development, implementation, and maintenance are very costly and it fails to provide the features of security and versioning. It is also too low-level and complicated and it has a steep learning curve. Therefore, CORBA objects have been hard to reuse effectively. A common major problem of the above mentioned solutions is that if the client has a firewall or a very restrictive server which only enables HTTP connections their communication could become impossible. As internet and complex distributed business applications evolves, the nature and scope of middleware technologies must change to integrate their applications with those that reside on heterogeneous platforms and to enable their applications to communicate and expose their services to global clients and partners. Thus, other technologies such as DDS, .NET Remoting, and Web services have quickly gained greater industry acceptance and support than predecessors like DCOM, RMI, or CORBA.

Web services have been designed to overcome the problems of platform dependence defining a standard way to exchange information through internet to an unprecedented level. Web services are based on open Web standards and broadly support interoperability across heterogeneous environments. These open standards make Web services indifferent to the operating system, object model, and programming language used. Web services are easily tunneled on HTTP in firewalls and proxies in contrast to DDS and .Net Remoting. However, it is not the ideal solution for all application models as it is not as efficient or reliable as DDS and .NET Remoting. NET Remoting is also limited to Microsoft Platforms, what makes it more appropriate on local networks. DDS is a powerful solution for real-time applications with high performance but might not be appropriate for remote laboratories applications for the following reasons: (1) it is restricted in firewall-enabled networks; (2) it relies on the Real-Time Publish Subscribe (RTPS) protocol which is not commonly supported in Web applications; and mostly important (3) it has only publish/subscribe support and doesn't support request/response communications. By applying performance enhancement techniques, Web services can be the only potential candidate for defining a LaaS paradigm.

Web services are either activity-oriented or resource-oriented. Activity-oriented Web services (i.e., also known as big Web Services) are represented by the communication SOAP and the specifications WSDL and Universal Description, UDDI. In addition to a variety of other associated specifications (i.e., referred as WS-*) that are maintained and supported by various bodies and entities. SOAP defines a standard way for exchanging XML-based information from one computer to another. It relies on any application protocol most notably HTTP—which can tunnel easily over existing firewalls and proxies—or Simple Mail Transfer Protocol (SMTP). WSDL is an XML-based file, which describes the available Web services, the data to be passed, and the methods for passing it in a standard way independently from the adopted programming language. Automated tools are available for programmers to generate a WSDL file from existing classes and vice versa. UDDI is used as a service broker which provides the potential for Web services to be registered in a central location, from where they can be discovered by service requestors. It enables requestors and providers to find each other based on: contact information and service available (white pages listing); taxonomies that are either standardized (yellow page listing) or user-defined; or types of

Web services they offer (green pages listing). Unlike WSDL and SOAP, UDDI has failed to reach widespread acceptance in the industry despite the fact that it has been mature and stable for several years now. Most developers prefer to build their proprietary registry instead and UDDI remains an optional extension to SOA. SOAP Web services are classified into two categories: SOAP RPC and document-centric SOAP Web service. In SOAP document-centric Web services, the requester and provider exchange SOAP messages after having agreed on the structures of the documents to be exchanged. The SOAP RPC Web service tunnels application-specific RPC interfaces through the generic SOAP interface. RPC interfaces require more complete and rigorous specification, and greater prior agreement. Effectively, they prescribes both system behavior—which is very difficult to prescribe in a distributed environment—and application semantics, and they are imperative rather than descriptive, which is not as interoperable as SOAP document-centric Web services and is contrary to the spirit of SOA. The topmost service orchestration layer determines the execution flows and provides companies with a language such as Web Services Business Process Execution Language (WS-BPEL) and Business Process Model and Notation (BPMN) to describe how data is exchanged within their interactions with trading partners in SOA context. In SOAP, data is always serialized as XML, which is considerably slower than other competing middleware technologies due to its verbose format. This may not be an issue when only small messages are sent. Techniques such as Message Transmission Optimization Mechanism (MTOM) are used to improve performance for the special case of XML with embedded binary objects. SOAP could be routed over HTTPS with either simple or mutual authentication. Furthermore, the WS-* specifications stack help to guarantee security, QoS, and reliability [86].

The other common form of Web services is the resource-oriented Web services, REST. REST is an architectural style that makes deliberate use of existing Web technologies. A resource is anything that has a Uniform resource identifier (URI) and may have zero or more representations. A representation is a self-descriptive document of a resource state that may use hypermedia to change application state—meaning that they may contain links to (representations of) other resources. REST is designed to use a stateless communication protocol, typically HTTP. Therefore, the server cannot hold any session state, and client requests necessarily contain all information needed to understand a request in isolation without referring to earlier requests. The HTTP protocol defines a uniform interface for accessing resources by a number of commands, which are Create, Read, Update, and Delete (CRUD) and are represented by the HTTP verbs: POST, GET, PUT, and DELETE, respectively. Two verbs are commonly used: GET for idempotent requests—meaning that duplicate consumer requests are processed once and only once—and POST for everything else. This is due to the fact that proxies and firewalls may not always allow HTTP connections that use any other verb. Also, POST and GET are the only two verbs that can be used in the method attribute of an Extensible HyperText Markup Language (XHTML) form. The most common resource representation is XML but other popular formats include JavaScript Object Notation (JSON) and XHTML. The interactions can be secured at the transport layer using Secure Sockets Layer (SSL) protocol while the messages can be secured using encryption and a digital signature. REST does not offer security features, encryption, session management, QoS guarantees, etc. but these can be added by building on top of HTTP. REST allows discovering Web resources without any discovery or registry repository. The newly proposed XML-based Web Application Description Language (WADL) provides standard descriptors for REST Web services just as WSDL for SOAP Web services. A WADL descriptor not only describes the service—including the grammars, resources, and methods—but also aids in the creation of stubs that are used to build service clients. Currently, tools for WADL that create stubs from descriptors are available only for Java environments. However, WADL is not yet widely supported. Version 2.0 of WSDL also offers support for REST and for all the HTTP verbs.

SOAP communication produces considerable network traffic and causes higher latency than other competing middleware technologies. Perhaps more importantly, the generation and parsing of SOAP messages can be computationally very expensive. In spite of the perceived complexity, SOAP have gained widespread adoption as the gateway technologies capable of delivering interoperability between heterogeneous systems owing to its transparency and independence. SOAP relies on any application protocol and support synchronous and asynchronous interaction patterns, albeit it might not always use Web-friendly protocols. From the perspective of REST web services, there is no choice but to use HTTP protocol and only synchronous interaction pattern is supported.

SOAP has a much more elaborate complex of standardization efforts, while the light-weight REST provides a simplified interface without using a lot of processor power and avoiding the need for various layers of the WS-* stack. In terms of programming, REST requires a lot of low-level coding while SOAP provides better tool support and programming interface convenience. REST allows resources to be represented by any format (i.e., HTML, GIF, PDF, etc.) while WS-* is limited to XML-based data type. A notable limitation in REST is that for idempotent requests, it is not possible to encode large amounts of input data in the resource URI, as the server will either refuses such requests or crashes [87]. In sum, REST Web services are preferred for hypermedia systems, e-commerce, and ad-hoc composition over the Web (mashup), while SOAP Web services are preferred in Enterprise application integration (EAI) and in sophisticated Business-to-Business (B2B) applications with a longer lifespan and advanced QoS requirements. SOAP is supported by application development tool makers such as International Business Machines Corporation (IBM), BEA Systems, Inc. (www.bea.com), and Microsoft, whereas REST has been preferably used by Amazon, Google, YouTube, and others to create interfaces to their Web services. A brief comparison between both solutions is provided in Table 9.

The simplicity and the high performance of REST—as well as its homogeneity with Web applications in general and mashup applications in particular—makes it the ideal candidate for implementing SOA. In our case service discovery wouldn’t be necessary since its role will be assigned to the market place. Web services, however, are only suitable for transactions or request/response messages.

For persistent connections like streaming of data (e.g., Webcam video streaming or measurement streaming), other approaches should be considered to allow server pushing like encoded over TCP or User Datagram Protocol (UDP) and the most recently HTML5 WebSocket APIs— which will help avoiding the reliance on any decoding plugins and promoting the development of Web native applications. WebSocket APIs haven’t been officially standardized by the World Wide Web Consortium (W3C) yet. Nevertheless, it is recommended to rely on low level protocols such as TCP and UDP to achieve maximum abstraction and compatibility. Thus, service description file should include laboratory/experiment information and all its supported functions and connections. Figure 40 provides an example of a service description file.

Table 9.A comparison between SOAP and REST.

	SOAP	REST
Transport protocol	any application protocol	HTTP
Communication pattern	synchronous and asynchronous	synchronous
Style	RPC and messaging	highly loosely-coupling RPC
Automated coding tools	supported	not supported
complexity	high	low
Service Discovery	supported (e.g., UDDI)	do-it-yourself
Service Description	inherently supported (e.g., WSDL)	supported (e.g., WADL)
QoS	supported (e.g., WS-*)	not supported
Service composition	supported (e.g., BPEL4WS)	supported (e.g., mashup)
Payload format	XML	XML, JSON, and others
Performance	low	high
SOA	supported	supported
Data size	unlimited	limited
Security	supported	supported

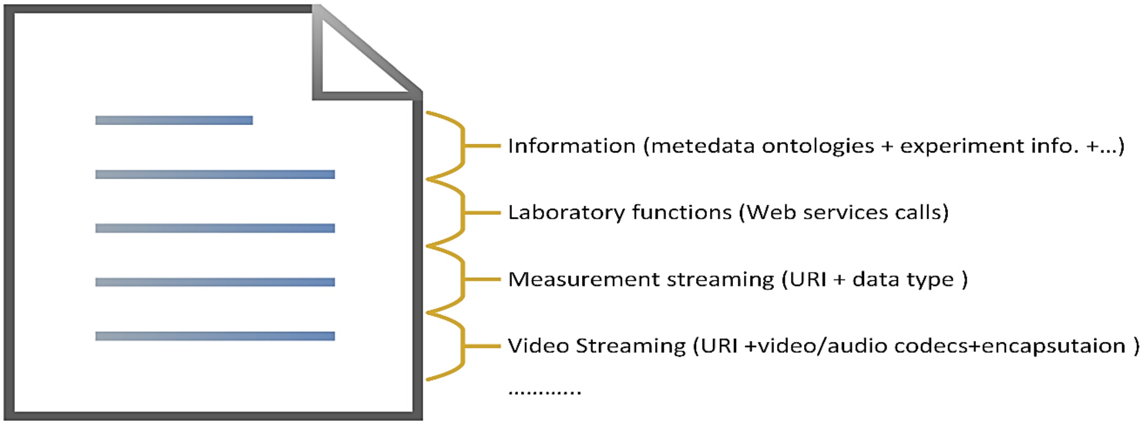

Information (metedata ontologies + experiment info. +…)
Laboratory functions (Web services calls)
Measurement streaming (URI + data type)
Video Streaming (URI +video/audio codecs+encapsutaion)
…………

Figure 40. Service description file.

5.7. Modular Architecture

Consider the generic[1] and common remote laboratory architecture shown in Figure 41. Typically a remote laboratory consists of a laboratory server—which is connected to all the equipment and hosts the control software—in addition to any combination of the on-demand components shown in Figure 47 [88]. The control software can be developed either from scratch using a multipurpose programming language (e.g., Java, C#, or C/C ++) or using a commercial solution, commonly LabVIEW or MATLAB. The DAQ board acts as an interface between the laboratory server and the equipment that don't support direct interface to the computer. A Webcam is used for video live streaming. Commercial Automatic Test Equipment (ATE) is used for specific signal generation or acquiring tasks. Standard connectors are used for connecting components directly to the laboratory server without intermediaries and they encompasses USB, LXI, PXI, and AdvancedTCA Extensions for Instrumentation and Test (AXIe). Sensors and actuators are used to convert physical parameters from the objects under control to electrical signals, and vice versa, respectively. A switching board is used for remote switching or wiring any terminals either from the objects under control or the ATE. Some applications might require a controller—in addition to the laboratory server—for a specific task. Commonly used controllers are either microcontrollers or FPGAs.

Modular remote laboratories are based on interchangeable component modules that expose their I/O terminals or their I/O connectors (i.e., if they physically don't exist or unavailable) in an independent and an abstracted way. Some components can be modularized and some are fixed and cannot be modularized or interchanged programmatically (e.g., laboratory server and DAQ board). The idea beyond modularizing remote laboratories is to facilitate maintenance, reusability, and interchangeability of components seamlessly and programmatically. In this sense, if the I/O terminals and connectors of all the component modules of a remote laboratory are provided in a "service description file" in order to allow consumers to get clues on them as shown in Figure 41, the consumer would be able to consume them separately. Furthermore, if one of the component

[1] This architecture was deduced from the study conducted in Chapter 2 and after analyzing different types of remote laboratories for different applications.

modules is not available and the appropriate I/O connectors are provided instead, the consumer could replace this module with his/her own one instead of the fully-reliance on the provider's equipment. For instance, a remote laboratory for image processing may expose an API to allow user to connect his/her camera capture. The image will be transmitted to the laboratory for processing and then return back to the user.

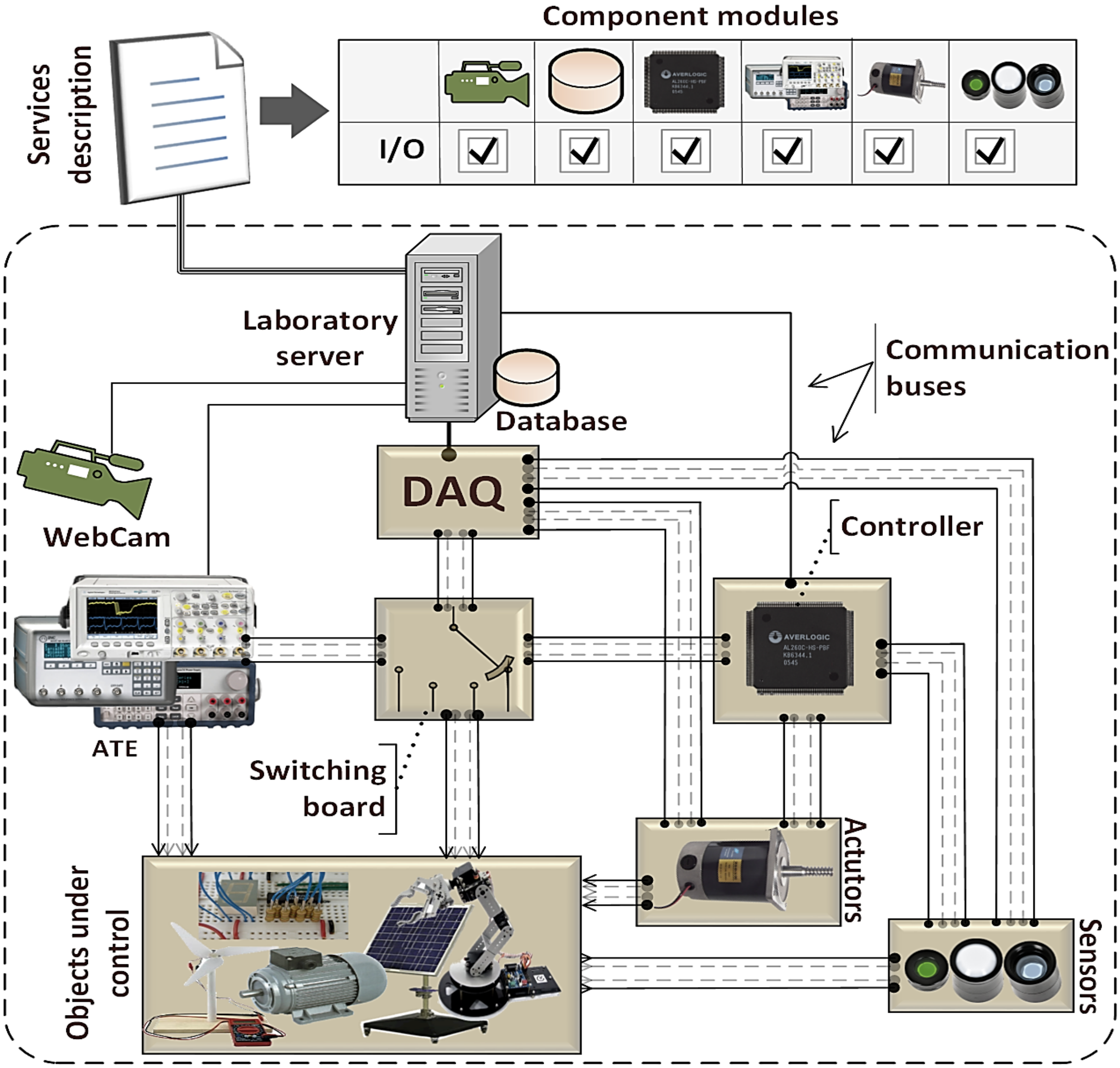


Figure 41. Modular remote laboratories architecture.

Each of the main components of the modular remote laboratories architecture shown in Figure 41 is going to be studied in details in the following subsections.

5.7.1. Data Acquisition (DAQ) Card

The DAQ card or board acts as an interface between the laboratory server and the equipment that don't support direct interface to the computer. It consists of modules such as A/D and D/A converters and signal conditioning circuitry—to convert sensor signals into a form that can be converted to digital values. In order to convert signals to the range suitable for the DAQ board, modules may be used such as current and voltage transformers, voltage and current dividers. In addition, to assure the complete isolation if of the electrical hardware and the DAQ board to voltage-to-frequency/frequency-to-voltage converters and optocoupler may be used. Commercial ATE—if required—are connected directly to the laboratory server without intermediaries. Sensors are used to convert physical parameters from the objects under control to electrical signals—which is transmitted to the laboratory server after being converted to digital values by the DAQ board.

5.7.2. Switching Board

The switching board is responsible for remote switching or wiring terminals of either the objects under control or the ATE according to the received digital input from the DAQ board. A switching circuit can be matrix-based or multiplexer-based as depicted in Figure 42(a) and Figure 42(b), respectively.

In the multiplexer-based switching, a certain terminal can be selected. In the matrix-based switching, the microcontroller receives the input digital signals and allocates the requested relays and closes/opens them accordingly. Relays are arranged in columns and rows to allow multiple connections. The microcontroller selects the corresponding columns and rows to realize the requested connection. Common relay types are electromechanical, reed, solid-state, and Field-Effect Transistor (FET).

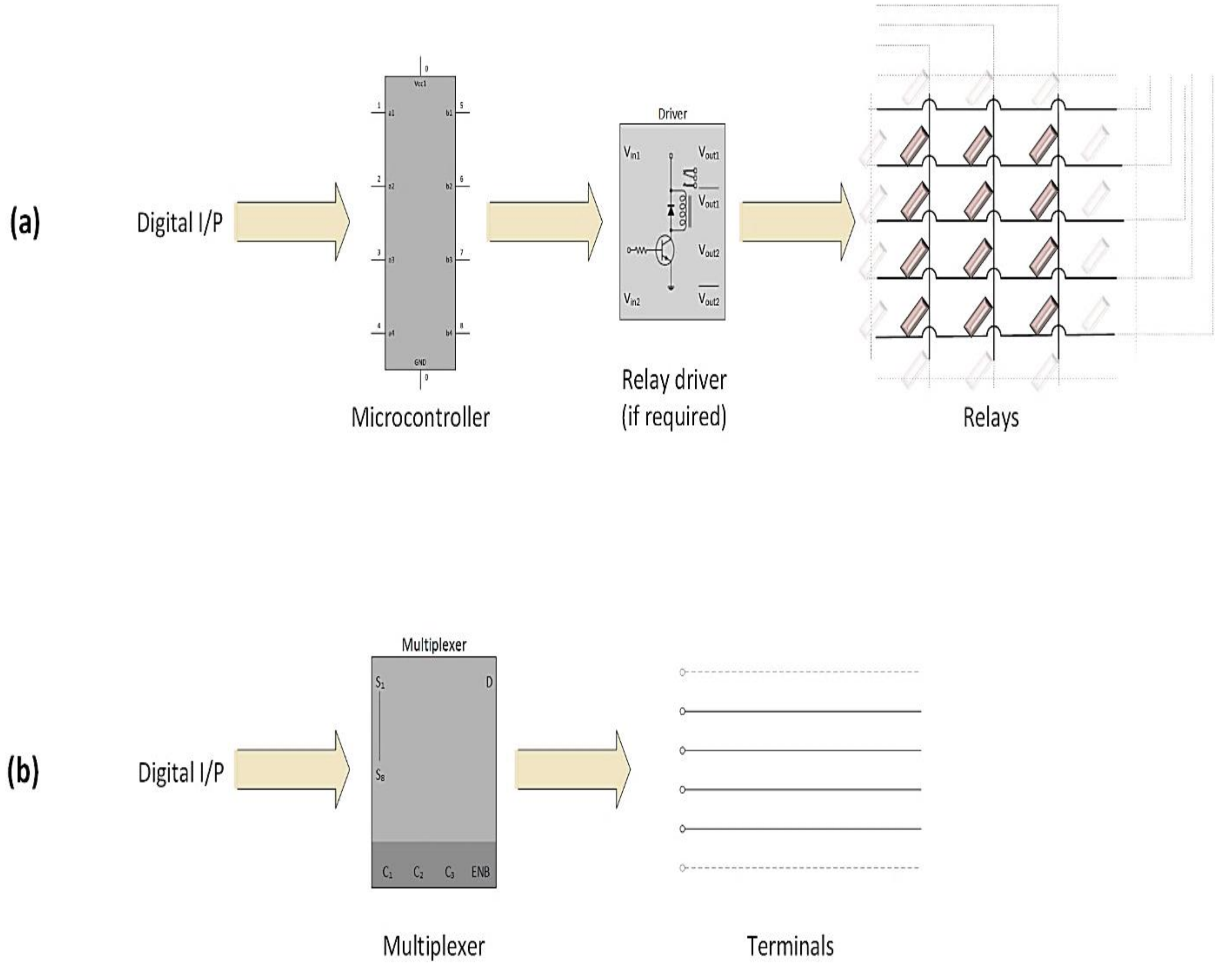


Figure 42. Topology of relay switching circuits.

Electromechanical relays are based on a spring-loaded conductive armature that is pulled by an energized coil magnet to connect with one contact or another. They support high currents (i.e., from 2 to 15 A) and provide low switching (i.e., typically 10 to 100 ms), which could be too slow for some applications. They come in many arrangements such as SPST, SPDT, Double Pole Double Throw (DPDT), etc. they can be activated by either AC or DC voltage. For example, when a DC coil is fed with an AC current, the armature flips back and forth as the polarity of the applied current changes, and when an AC coil is fed by an AC current the armature is pulled towards one switch contact and is held in place as long as the current is applied. Electromechanical relays may have a latching feature, meaning that when a pulse is applied the switch closes even when the pulse is removed and remains closed by a permanent magnet till the next pulse is applied. Latching relays are preferable for low-voltage applications because the lack of coil heating minimizes the EMF. Since they have moving parts, they have a predefined mechanical lifetime (i.e., typically 1 million cycles) and they should be replaced before reaching this value to avoid measuring errors. Voltage spikes resulting from the inductive behavior of the coil can damage the switch contacts, thus transient suppressors are recommended. A transistor driver circuit is used when the current and voltage requirements for relays exceed those available by the source. It is used then to drive the relay's coil and acts as a current-flow controller.

Reed Relays are made of wire coil wrapped around an air-tight glass container that encapsulates a pair of ferromagnetic reeds (i.e., thin and flexible metal strips that have contacts on their overlapping ends). Whenever a current is sent through the coil and energizes it, the reeds are forced together by a magnetic field, thus closing the switch. When the coil is de-energized, the spring force in the reeds pulls the contacts apart. Unlike electromechanical relays, reed relays are typically limited to SPST switching. Reed relays are designed for moderate currents (i.e., typically 500 mA to 1 A) and fast switching (i.e., around 0.2 to 2 ms), owing to their small contacts. The lifetime of reed relays is much higher than an electromechanical relay. Reed relays are protected from system capacitance inrush currents using a series impedance such as a resistor or ferrite. Reed relays are not preferred for very low-voltage applications owing to their higher thermal EMF caused by their ferromagnetic materials and its resulting high noise in measurements.

Solid-state relays are made of a photo-sensitive solid-state electronic switching device, such as transistors or thyristors, and a control circuitry (a LED). They are based on optical coupling; the LED is energized by a control voltage to turns on the switching device which, in turn, switches the load—thus the control circuitry is electrically isolated from the load. Solid-state relays come with a wide range of current ratings (i.e., from few mA up to 100 A) and have extremely fast switching speeds (i.e., typically 1 to 100 ns). Because there are no mechanical parts, they have infinite mechanical lifetimes. Solid-state relays are useful for high-voltage applications because of its optical coupling feature. Solid-state relays are also limited to SPST switching and are susceptible to surges currents and may be damaged when used at signal levels above their rating in power.

FET relays they are based on a series of CMOS switching transistors. Since gates are driven directly from the control circuitry without LED and there are no mechanical parts, they are the fastest of all switches. However, they lack physical isolation barriers and thus may only be used with low-voltage signals.

Commercial switching board solutions are also available from companies such as NI (http://www.ni.com/switches/) with varying parameters (i.e., relay type, maximum voltage and current range, and switching speed).

5.7.3. **Controller**

The controller is an additional processor used for implementing a certain control strategy such as PID control, leaving monitoring and changing parameter tasks to the laboratory server. Commercial controller boards support direct interface to the computer. The controller receives feedback from sensors and provides output though actuators or any other peripherals. It may also send control commands to the switching board. Controller boards can be implemented by either a microcontroller, a PLC, or an FPGA.

A commercial µC board consists of a CPU, volatile and/or nonvolatile memory, support for communication standards such as USB, support for boundary scan using JTAG, and peripherals such as switches, relays, solenoids, LEDs, LCDs, RF devices, sensors, D/A converters, A/D converters, timers, motors, and PWMs. Microcontrollers are programmed in either assembly language or high level programing languages such as C. The IDE includes tools that let users edit, assemble, compile, run, and debug their code to the microcontroller. Microcontrollers are commercially distributed in 8, 16, and 32 bit.

A PLC is a specific microcontroller adapted for industry applications. It is armored for severe industrial environment conditions. It is programmed in ladder logic, which is conceived to be easy to grasp by technicians. It can also be programmed in other variety of ways such as specially adapted dialects of BASIC and C. It is commonly used for implementing control strategy in automation-related high voltage laboratory applications because of its programming simplicity and in order to teach students ladder logic programming. It supports standards for interfacing hardware (e.g., Profibus), and software (e.g., OPC UA).

FPGA it is a PLD programmed in HDLs (e.g., Verilog or VHDL). It consists of configurable logic blocks, I/O blocks and programmable interconnections. FPGAs can have ten thousands of logic blocks in addition to memory and other resources. Each configurable logic block is made of smaller logic modules which can be configured combinational logic, registered logic, or a combination of both. A logic module can produce a Sum-of-Products (SOP) function with up to16 product terms. Density can range up to 180, 000 logic modules and over 1000 input and output pins. A hard core portions— that cannot be programmed by the customer—may be added by the manufacturer such as processor core, DSP, or memory. They occupy less of the available capacity than if the user programmed them (i.e., specific embedded functions programmed by customers are known as "soft-core"). Most PLDs are complaint with IEEE Std. 11491.1 and supports boundary scan, bypass, and instruction registers. FPGAs provide high flexibility, high capacity, and can implement a full SoC. Commercial FPGA boards come with communication standards interfaces and peripherals such as LEDs, push buttons, LCD display, A/D and D/A converters, EPROM, RAM, and clock generator.

5.7.4. **Automatic Test Equipment (ATE)**

ATE can be either stand-alone or modular instruments as shown in Figure 43.

Stand-alone instruments are connected directly to the server as if they were peripherals. Components of each instrument, including the processor, are put together in a single box as a discrete entity. The firmware of stand-alone instruments is uploaded and updated only by the manufacturer and it is used to enable users to interact with the instruments' front panel. Optionally, users could control the instruments via a developed control software installed at his/her PC.

Communication buses that links the instruments with other systems are commonly GPIB and LXI and lesser extent RS-232 and USB. Each of these buses have its own weakness and strengths, which makes some more suitable for applications than others. For instance, GPIB is widely available among a variety of instrumentation, USB is widely available and easier to connect, and LXI is useful for distributed and distance applications. User can select among this variety of available instrumentation buses based on factors such as bandwidth, latency, performance, cost, and flexibility. While stand-alone instruments can be beneficial in terms of flexibility, these instruments face limited integration and expandability. Stand-alone instruments duplicate the power supply, chassis, and controller for every instrument. This increases cost and size and decreases reliability.

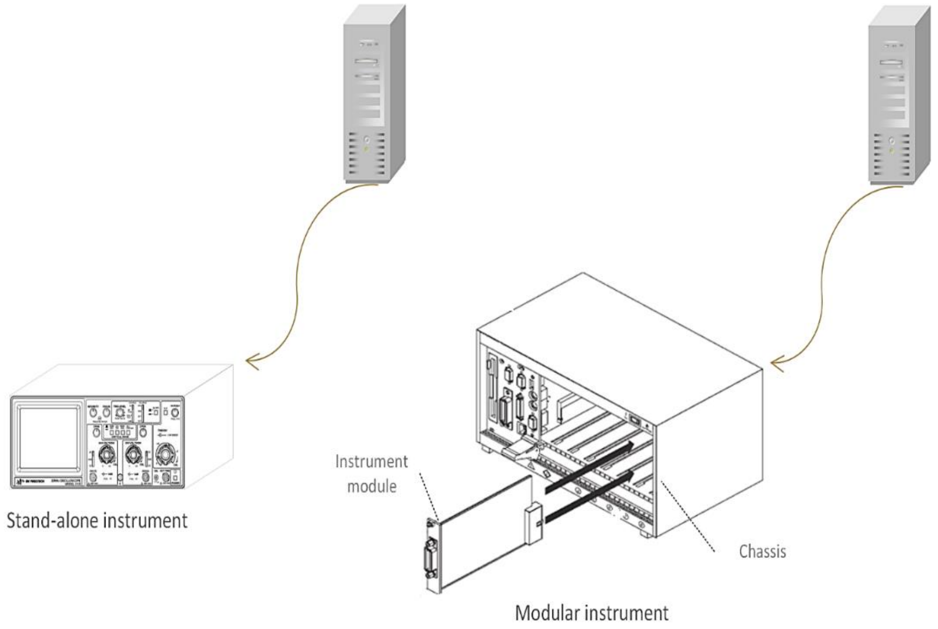


Figure 43. Stand-alone and modular configurations of ATE.

Modular instruments shares a controller and many of the components such as chassis and power supply across instrument modules instead of duplicating these components for every single instrument. The instrument modules are embedded in a chassis or an integral instrumentation platform and all of them are connected to the communications bus that puts them in contact with a single processor. Components and instrument modules from multiple vendors can be easily integrated into one system and system can be scaled as needed. In some cases instrument modules are simply a peripheral that can optionally be connected directly to the host computer. In this case, the host PC provides the chassis and the power supply. Beside the hardware-level flexibility, modular instruments relies on software-defined virtual instruments, which allows users to build and modify their own custom measurement platform using emerging standards such as VISA and IVI. While, user-defined software can, as well, be applied to stand-alone instruments, it is ideally paired with general-purpose and modular systems, where the full flexibility and performance of the measurement software can be exploited. Key elements of using modular instruments are: cost and size reduction through a shared chassis, backplane, and processor; greater throughput through a high-speed connection to the host processor; and greater flexibility and longevity through user-defined software. Modular instrumentation platforms are built with communication buses such as PXI and VME eXtensions for Instrumentation (VXI). These platforms also may also support interfaces to other buses such as GPIB or Local Area Network (LAN). Systems built on modular instruments can be up to 100 times faster than systems built solely on traditional instruments.

5.7.5. Communication Buses

Communicant buses plays an essential role in the performance of the LaaS architecture and the compatibility between its components. Relying on well-known standards—or even using connectors—will ease maintenance and upgrades by requiring only minor changes in specific layers as opposed to redesigning the system to accommodate new components. This will facilitate reusing and exchanging equipment and mitigate cost. Factors to be considered when selecting a standard communication bus includes bandwidth, distribution, connector ruggedness, and cable range. The widely accepted standards for equipment connections are RS-232, USB, GPIB, VXI, Peripheral Component Interconnect (PCI), PXI, AXIe, and LXI [89, 90].

RS-232 is a serial bus where single bit of data is transmitted at a time. It has been for a long time the primary standard for computer serial ports. While the standard recommends a 25-pin connector, 9-pin connectors are common, and a three-wire arrangement is often used. Using low capacitance cables, it is possible to maintain full speed communication over distances up to 300 meters. Baud rating according to the standard can reach 20 K, however modern devices can operate much faster. Its main disadvantage is its relative low speed. As a result, it was deprecated and replaced by much faster serial buses such as USB and Ethernet connections, but it is still found in many devices. Connectors such as Universal Asynchronous Receiver/Transmitter (UART) are used to exchange data between serial and parallel buses.

USB is a serial bus that transmits data on a single pair of wires. It supports all kinds of data including video and audio. In the recent years, it has become the most popular and widespread for connecting devices to computers and for stand-alone instruments due to its ubiquity on computers and its seamless plug-and-play and high-bandwidth capabilities. A single host controller can support up to 127 peripheral devices and bandwidth is distributed among them. Unlike other busses such as GPIB and LAN, USB is auto-detected and easily recognized by the PC when connected. Compared to RS-232, USB is smaller in size and much higher in speed but its cable length is limited to 5 meter long. The data transfer rate of the High Speed USB or USB 2.0 is up to 60 MB/s and the Super Speed USB or USB 3.0 is approximately 596 MB/s. Connectors of USB are the least robust and secure and may need ties in case of sensitive applications to keep them in place. The USB Test and Measurement Class (USBTMC) specification builds on top of USB in order to allow its complaint USB devices to adopt VISA standard.

GPIB or IEEE 488.2 is an 8-bit parallel electrical bus specifically designed for Test and Measurement (T&M) applications and stand-alone instruments. It is supported by most off-the-shelf measuring instruments and it has been the first choice for ATS since its birth. Compared with other buses, GPIB has a longer life cycle and lower latency. The bus uses 24-pin connector and employs 16 signal lines—8 bi-directional for data transfer, 3 for handshake, and 5 for bus management—and 8 ground return lines. The maximum transmission rate is 1.8 MB/s and up to 15 devices are allowed to share a single physical bus of up to 20 meters total cable length in a daisy chain. Converters are available, such as the NI PCI-GBIB and GPIB-USB, to facilitate interfacing GPIB with other buses. The Standard Commands for Programmable Instruments (SCPI) is an additional layer to IEEE-488 which specifies standard set instrument commands to be used by different models and manufacturers. SCPI can also be used with other buses such as RS-232, USB, and Ethernet.

VXI or IEEE 1155 is a modular instrumentation platform based on the Versa Module Europe (VME) bus. It is used for building complex T&M and data acquisition applications through a reconfigurable combination of instrument and component modules. It is composed of a mainframe chassis, which supports up to 13 slots. One slot is reserved for the controller and the other 12 are dedicated to the instrument modules. VXI uses the full 32-bit VME architecture and has a maximum bandwidth rate of 40 MB/s. VXI system can be controlled by an external controller (i.e., a general purpose PC) by linking to its backplane via GPIB-VXI interface module or high-speed Multisystem eXtension Interface (MXI) connector. An MXI-connected PC is functionally equivalent to an embedded VXI controller and both provides potentially dramatic performance improvements over GPIB. MXI cable could be up to 20 meters and up to eight MXI devices can be daisy chained on a single MXI cable.

PCI is a high-performance internal bus which provides the lowest latency and the highest data throughput or bandwidth among all the available T&M buses. It is one of the most famous and widely adopted PC buses of all time. PCI provides up to 132 MB/s of bus bandwidth. This bandwidth is shared across all devices on the bus. PCI allows 32- and 64-bit data transfer. Unlike other buses which are directly used to cable external instruments, PCI is an internal bus implemented as carrier bus within plug-in cards (CompactPCI cards) and modular instrumentation systems such as PXI. When connected to a PXI system, it can be extended up to 200 meter via NI fiber-optic MXI connectors. Since PCI is an internal bus, its robustness is constrained by the stability and the ruggedness of the system it which it resides. PCI is a plug-and-play bus and its modules are automatically detected once PC is booted. PCI Express (PCIe) is the latest evolution of the PCI standard. PCI Express is a higher bandwidth bus and it is the only bus that gives dedicated bandwidth to each peripheral device instead of dividing bandwidth across the connected peripherals. Unlike PCI, PCIe uses independent data lanes and each lane can scale up to 250 MB/s—with a maximum of 16 data lanes and 4GB/s total bandwidth—and achieves complete software backward compatibility with PCI.

PXI is a modular instrumentation platform similar to VXI but adopts the PCI bus as an underlying bus architecture, which, in turns, enables a higher communication speed. PXI extends CompactPCl and maintains complete interoperability with CompactPCl products. Similar to VXI but in a significant smaller size, a PXI system is composed of a chassis, instrument and component modules that fit into the chasses' slots, and a controller. Total shared bandwidth could reach up to 132 MB/s. Using MXI connectors, PXI systems can be controlled by external PCs and all peripheral modules in the PXI system would be recognized as PCI devices once PC is booted. It is also possible to implement either a daisy-chain or a star topology to build multi-chassis systems via MXI. A number of methods are available to build hybrid systems interfacing other technologies such as USB, LXI, and VXI. PXI Express (PXIe) is a newer version which takes the advantage of PCI Express technology. PXI Express increase the available PXI bandwidth from 132 MB/s to 250 MB/s (per lane) while maintaining software and hardware backward compatibility with PXI modules.

The new AXIe leverages existing standards from Advanced Telecom Computing Architecture (AdvancedTCA), PXI, LXI and IVI. It is easily integrated with other platforms and offers a higher performance, greater scalability, and more flexibility than PXI and PXIe. AXIe tackles some of the constraints of PXI and PXIe, most notably the module size and power limitations. AXIe allows up to 900 cm^2 module board space and 2700 cm^3 volume. An AXIe chassis can contain up to 14 vertically-arranged slots or up to 16 horizontally-arranged slots. It is well-suited to high power applications of up to 200 W per slot. AXIe supports module connectivity for other standards such as LXI, PCIe, and PXIe.

LXI is a LAN-based standard for T&M systems. It defines a specification for connecting ATE using Ethernet and adding optional triggering and synchronization features. LXI offers integration advantages of modular instruments without the constraints of card-cage architecture. It can be used at any level of network complexity ranging from stand-alone instrument to multi complex-modular-systems operated remotely through Internet. 100BASE-TX (fast Ethernet) and 1000BASE-T (gigabit Ethernet) have a theoretical shared maximum bandwidth of 12.5 MB/s and 125 MB/s, respectively. LXI takes advantage of Ethernet's ability to operate at unlimited distance using switches, routers, hubs, or even optical fiber cables. LXI is ideally suited for distributed systems and remote monitoring owing to the innate distribution of LAN. This allows instruments to be distributed across the network without geographical constraints. LXI devices are not automatically configured by the PC and should be manually configured using their assigned IP address and subnet. Data transferred throughout the network can be secured using VPN. LXI standard defines a core class incorporates a Web browser via an Ethernet port as well as IVI driver in addition to optional extended functions that bring capabilities such as synchronization based on IEEE 1588 Precision Time Protocol (PTP) and precision triggering based on LXI trigger bus. Table 10 provides a comparison between the above mentioned busses.

Table 10. Comparison between different communication buses.

	Bandwidth	**Latency (µs)**	**Multi-device**	**portability**	**Distributed measurement**	**Connector robustness**	**Range (without extender)**
RS-232	20K bit/s	60	*	***	*	**	15m
GPIB	1.8 MB/s	50	*	***	*	****	20 m
USB	596 MB/s	90	**	****	***	*	5 m
VXI	40 MB/s	0.4	***	*	**	**	Internal PC bus
PCI	132 MB/s (shared)	0.6	***	*	**	**	Internal PC bus
PCIe	250 MB/s (per lane)	0.8	***	*	**	**	Internal PC bus
PXI	132 MB/s (shared)	0.6	****	**	***	***	Internal PC bus
PXIe	250 MB/s (per lane)	0.8	****	**	***	***	Internal PC bus
AXIe	2 GB/s (per lane)	0.4	****	**	***	***	Internal PC bus
LXI	125 MB/s (shared)	1000	**	****	****	**	100 m

When evaluating bus performance for ATE, bandwidth is the most important factor to be considered. It measures the ability of the bus to transfer an amount of data in a given period of time. Bandwidth can be either dedicated to a single device or shared among several devices. Delicate remote laboratories applications such as signal processing would require a faster bus with higher bandwidth. Latency is another important factor that determines the delay in transmitting data across the bus. For example, in a PID control system, latency can directly impact the maximum speed of the control loop. As demonstrated in Table 10, each bus is unique and have different strengths which might makes it more suitable for certain applications than others. For example, GPIB has the widest availability of instrumentation and adoption in ATE and therefore it is useful for equipment reuse. GPIB support greater bandwidth and cable range than RS-232. USB provides wide availability as well, easy and auto-detecting plug-and-play connectivity, and higher throughput than GPIB. Card-cage standards such VXI, PXI, and AXIe belong to the same family and provides the highest bandwidth and performance and the lowest latency. They add the industry's best triggering, timing, and synchronization capabilities and used for critical applications that demands a modular solution which is expected to reduce cost and size and prove higher performance and flexibility. AXIe is by far the most advantageous in terms of performance, scalability, and flexibility. It also support a greater power and module size range but it is less used because it’s a relatively new standard. Despite the lower performance of LXI, it is well-suited for distributed systems and has a clear advantage over other buses in life cycle and distribution.

In complex remote laboratories applications however, no single bus technology would meet all needs. Thus, standard communication busses are likely to continue to coexist within hybrid systems, since each standard has its unique strengths and weaknesses. In hybrid systems, users take advantage of various test platforms and integrate multiple platforms into one system by combining modular instrumentation buses such as PXI and VXI with peripheral buses for stand-alone instrumentation such as GPIB, USB, and Ethernet/LAN. By this way, a system would combine the software flexibility and high throughput provided by modular instrumentation buses with the specialized functionality that might be provided by a stand-alone instrument. The hybrid system diagram of Figure 44 gives one example of a hybrid topology.

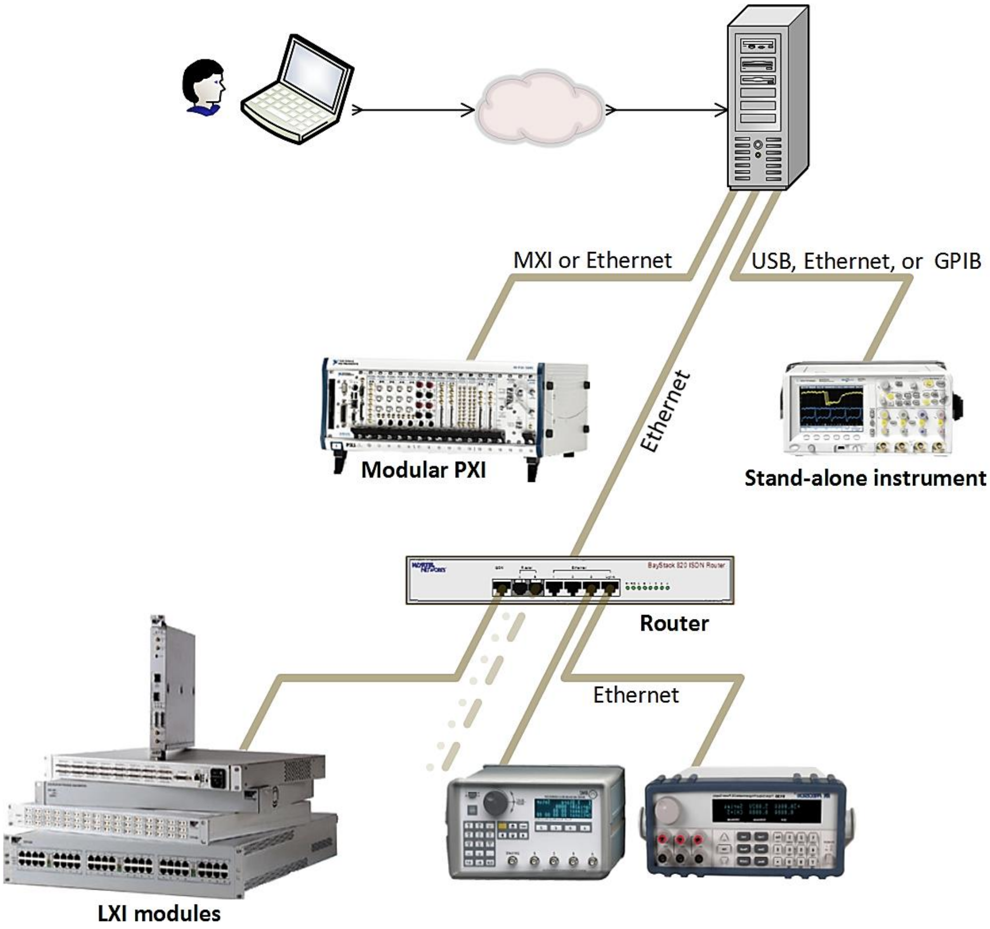


Figure 44. Topology of a hybrid bus system.

The key to ensure a successful develop of a hybrid system or an interoperable system—that will facilitate reusing and exchanging equipment guaranteeing ease of maintenance and upgrades—is to employ drivers with interoperable APIs—that are neither tied to a particular piece of hardware or vendor. Taking advantage of the abstraction provided by standards such as VISA and IVI, the software layer could adapt and keep up with changing commercial buses, which in turn, allows best code reuse, modularity, and longevity.

VISA is an industry standard implemented by several T&M companies that provides a common I/O API to communicate with the instrument driver software independently of the instrumentation bus used. VISA delivers a standard set of function calls which significantly reduces the time and effort involved in programming different I/O interfaces. By this way, user's investment in a software will be preserved even if he/she migrates to a new bus. Yet VISA simplifies instrument control programming significantly, users still need to acquire a low level knowledge of instrument communication and function calls. To address this issue, instrument and software vendors started releasing instrument drivers which abstract instrument functions in an appropriate way to be easily called by users. Drivers allow users to configure an instrument to perform a set of synchronized tasks with a single high-level software function which in addition to providing a common interface to the user, drivers simplify instrument control and significantly reduce the amount of time required to develop an instrument control software application.

Standard instrument drivers based on IVI enables interchangeability of instruments from a same class regardless of their manufacturer or bus type. The IVI specification defines an open architecture for a set of classes of instruments and a set of common software components known as IVI shared components. IVI shared components are common to all implementations and provide common services to all drivers and driver clients. The current 13 instrument classes defined are: DMM, oscilloscope, arbitrary waveform/function generator, DC power supply, AC power supply, switch, power meter, spectrum analyzer, RF signal generator, upconverter, downconverter, digitizer, and counter/timer. Additional class specifications are currently under developing. The goal of IVI is to cover 95% of the models in a given instrument class in their base capability group. Instrument classes break instrument functionality into capability groups. All IVI drivers communicate to the instrumentation hardware through an I/O library that might be VISA (i.e., particularly for VXI and GPIB). The minimum functionality that an IVI instrument driver must implement is defined in the fundamental capability group while the optional extended functionalities are defined in the extension groups. There are three types of IVI drivers and each with different levels of interchangeability: (1) IVI class drivers; (2) IVI class-compliant specific drivers; and (3) IVI custom specific drivers. IVI class drivers contain all of the base and extended capabilities for instrument classes. They use completely generic function calls that are not tied to a specific driver library. IVI class-compliant specific drivers comply with IVI class specifications but also describe the low-level driver calls in addition to the functions and attributes for the instrument class. Therefore, an IVI class-compliant specific driver of two different instrument models might be incompatible. In order to allow interchangeability, the developed program software must make generic function calls and avoid low-level driver calls. IVI custom specific drivers don't conform to any of the IVI class specifications and do not allow for hardware interchangeability. They are written for an instrument for which there is no class specification but the driver author still wants the other benefits of the IVI architecture. For each of these driver types, the specifications describe both an ANSI-C API (IVI-C) driver and a COM API (IVI-COM) driver. IVI-C driver is presented in the form of Dynamic Link Library (DLL) file which contains C functions. Owing to the maturity and the stability of the C language, IVI-C drivers can be used in any platform. The IVI-COM drivers provide a standard COM interface to facilitate development in environments such as Visual studio and LabVIEW. Both formats, IVI-C and IVI-COM, can be used in the same application.

5.7.6. **Laboratory Server Software**

A key element in the modular remote laboratories architecture is the control software or the laboratory server software, which is hosted in the laboratory server. This software plays a very important role in the architecture design for two main reasons: (1) it is in charge of the direct control of equipment and communication between the laboratory equipment and the laboratory server; and (2) it provides the service description file, to be used and consumed by the users and wrapped in any container, system, or client as shown in Figure 41.

The laboratory server software could be developed either from scratch using a multipurpose programming language (e.g., Java, C#, or C/C ++) or using an IDE such as LabVIEW and MATLAB/Simulink (www.mathworks.com/products/matlab/). In the former solution, the drivers of the equipment is imported in the programming code to allow controlling the equipment through the code. The former solution is barely reported in the literature and is discarded in the LaaS paradigm because, in contrast to the principle objectives of this paradigm, it is very difficult to be adopted—especially in complex remote laboratory applications—since it requires a very high programming skills and learning curve. Also, the code written for an application cannot be reused in another application. The latter solution—particularly using LabVIEW or MATLAB—is widely adopted in almost all the developed remote laboratories. A survey is conducted on the adopted laboratory server software technology in all the Remote Engineering & Virtual Instrumentation (REV) conference (www.rev-conference.org) publications since 2008 till 2013. The REV conference is the only conference so far dedicated exclusively for remote laboratories and their related topics. Therefore, it is a viable scale to rely on. The results of the survey are tabulated in Table 11.

Table 11. Market share of laboratory server software.

	2008	**2009**	**2010**	**2011**	**2012**	**2013**
LabVIEW	17 *80.10%*	17 *94.44%*	12 *92.30%*	23 *92%*	26 *96.30%*	11 *100%*
MATLAB	3 *14.30%*	0 *0%*	0 *0%*	0 *0%*	0 *0%*	0 *0%*
Other	1 *5.60%*	1 *5.56%*	1 *7.70%*	2 *8%*	1 *3.70%*	0 *0%*

As clearly demonstrated in Table 11, LabVIEW dominates the market share in remote laboratory developments (i.e., the reasons will be discussed later in this chapter) and the number of applications relied on LabVIEW is increasing relatively over the years.

> *i.e., the last REV conference held in Australia in 2013 recorded a significant low participation number due to logistical and financial reasons with only a total number of 29 contributions, compared to approximately 65 in previous years.*

A graphical chart of the results in percentages is presented in Figure 45.

Even though these results reveal that LabVIEW is a de facto tool for lab server software development, MATLAB will continue to exist in remote laboratory developments owing to its powerful computing language for control algorithm development and simulation. MATLAB is a high-level technical computing language and interactive environment for high performance intensive numerical computation, algorithm development, data visualization, and data analysis. It is used in a wide range of applications, including signal and image processing, communications, control design, test and measurement, financial modeling and analysis, and computational biology. Simulink is an environment for multi-domain simulation and model-based design for dynamic and embedded systems. It is fully integrated with MATLAB and provides an interactive graphical environment and a customizable set of block libraries that allows users to design, simulate, implement, and test a variety of time-varying systems, including communications, controls, signal processing, video processing, and image processing. MATLAB is designed to be scalable to allow concurrent tasks of simulation and compilation. However, the real-time control on physical equipment cannot be performed by more than one user at any time.

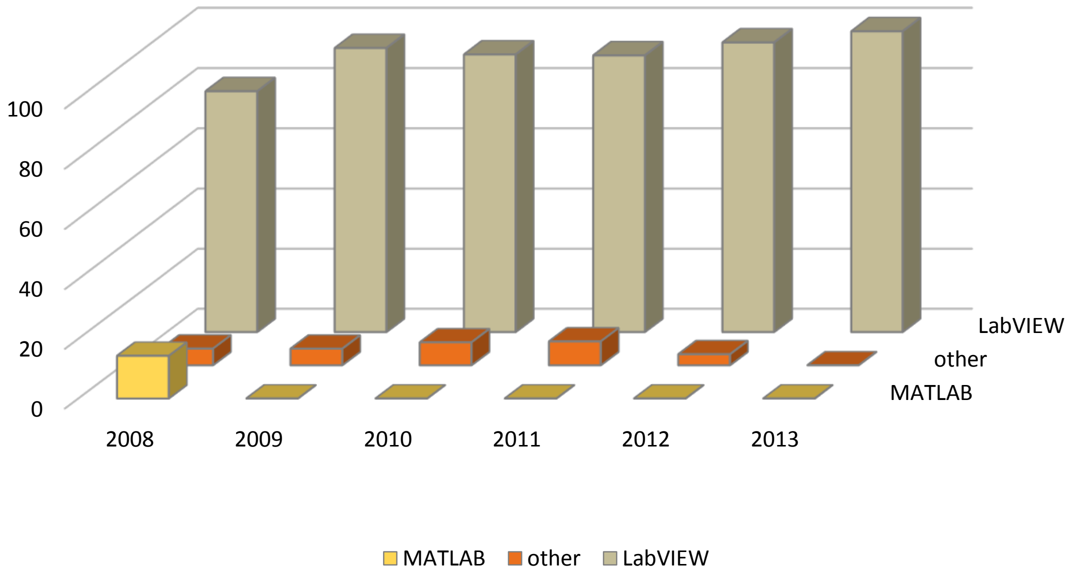


Figure 45. Market share of laboratory server software (chart).

Add-on toolboxes (collections of special-purpose MATLAB functions) extend the MATLAB environment to solve particular classes of problems in these application areas. MATLAB provides five main features for developing remote laboratories:

- **Data acquisition:** MATLAB can be connected to a variety of DAQ boards, from multiple vendors and via industry-standards such as USB, PCI, and PXI, through functions provided by the *Data Acquisition Toolbox*. The toolbox provides functions to access four subsystems commonly found on DAQ hardware: analog input, analog output, digital I/O, and counter/timer. In addition, the I*nstrument Control Toolbox* provides functions for connecting MATLAB directly to instruments (e.g., oscilloscopes, function generators, signal analyzers, power supplies, and analytical instruments) based on commonly used communication buses (e.g., I^2C, PXI, LXI, VXI, GPIB, and AXIe) via instrument drivers such as IVI and VISA or via text-based SCPI. The toolbox provides built-in support for TCP/IP, UDP, and Bluetooth serial protocols for remote communication with other computers and devices.
- **Integration with external software:** MATLAB applications can be compiled as a C/C++ libraries (i.e., DLLs in Windows or shared libraries in Linux and UNIX), using the *MATLAB Compiler*. ANSI/ISO compliant generation of standalone portable and readable C/C++ code from MATLAB code and Simulink code is also possible using *MATLAB Coder* and *Simulink coder*, respectively. Using *MATLAB Builder NE*, MATLAB code can be encrypted into .NET and COM wrappers to generate .NET and COM components that are callable from COM-compliant platforms and Common Language Specification (CLS)-compliant languages, including C#, F#, VB.NET, or ASP.NET. Similarly, Java classes can be created using *MATLAB Builder JA*. All the above mentioned solutions allows MATLAB applications to run outside the MATLAB platform, using a runtime engine called the MATLAB Compiler Runtime (MCR), and to be integrated in external software applications.
- **Programming embedded devices:** with respect to the software, *Embedded Coder* extends *MATLAB Coder* and *Simulink Coder* and generates readable, compact, and fast C/C++ code for use on embedded microprocessors and on-target rapid prototyping boards. The code can be executed with or without a RTOS and in single-tasking, multitasking, or asynchronous mode. *HDL Coder* generates portable and synthesizable Verilog and VHDL code from

MATLAB functions, Simulink models, and Stateflow charts. The generated HDL code can be used for FPGA programming or ASIC prototyping and design. With respect to the hardware, *xPC Target* enables executing Simulink and Stateflow models on a target computer for rapid control prototyping, Hardware-in-the-Loop (HIL) simulation, and other real-time testing applications. *xPC Target Embedded* Option is an extension to *xPC Target* that enables applications generated with *xPC Target* to run on a target computer in a way that it automatically loads and runs at startup, without needing a connection to a host computer.

- **Programming PLCs: w**ith respect to the software, *Simulink PLC Coder* implements control system models comprising feedback loops, mode and state logic, and math-intensive algorithms to be compiled and deployed in PLCs. It generates hardware-independent IEC 61131-3 structured text—in PLCopen XML and other file formats supported by widely used IDEs—from Simulink models, Stateflow charts, and MATLAB functions. With respect to the hardware, *OPC Toolbox* provides a connection to OPC servers allowing communication with devices that conform to the OPC such as SCADAs, and PLCs.
- **Database connectivity:** MATLAB can communicate with Relational Database Management Systems (RDMSs) through the *Database Toolbox*. The toolbox allows using SQL commands to read and write data and supports Open Database Connectivity (ODBC)-compliant and Java Database Connectivity (JDBC)-compliant databases, including Oracle, MySQL, Sybase, Microsoft SQL Server, and Informix.

Figure 46 provides a block diagram of the five main components of MATLAB for remote laboratory development.

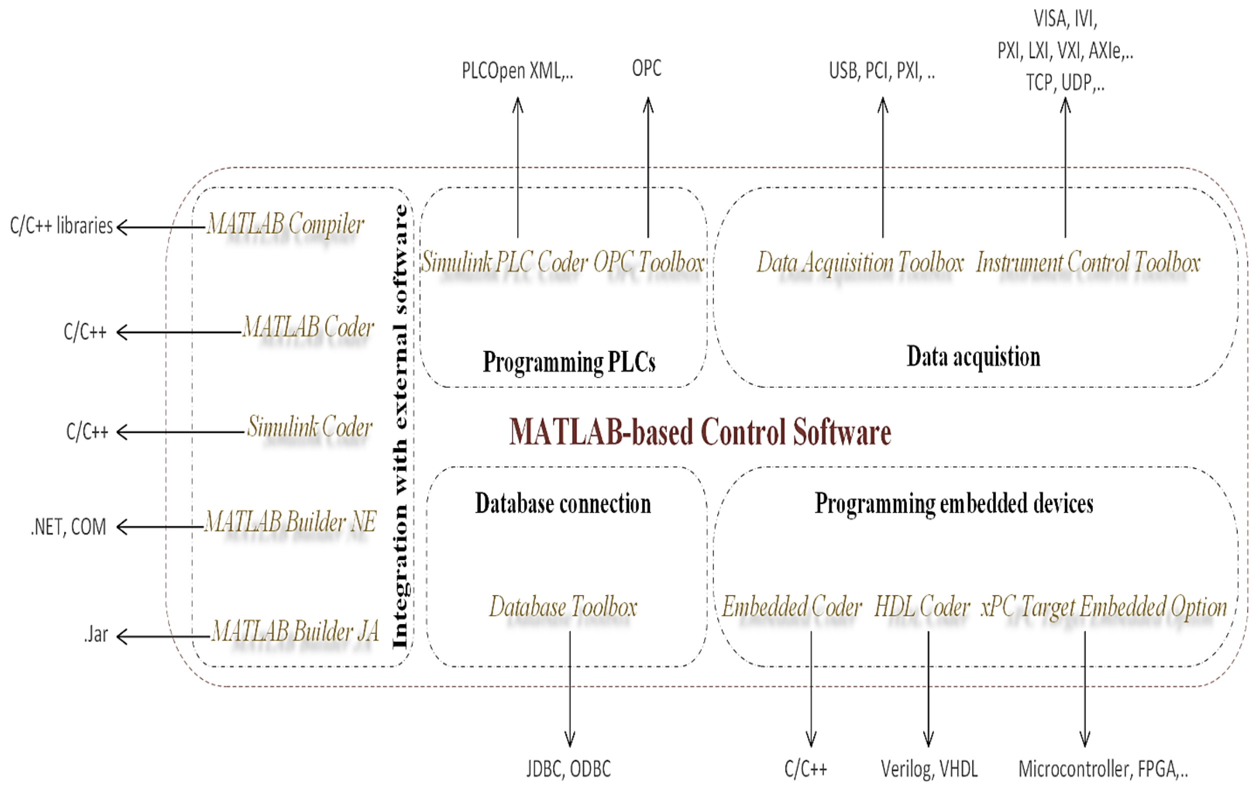


Figure 46. MATLAB's main components for remote laboratories development.

The most powerful and prevalent technology so far for lab server software development is LabVIEW. LabVIEW is an industrial leader graphical programming environment for developing, testing, and controlling systems using intuitive graphical icons, known as Virtual Instruments (VIs), which imitate the physical instruments. It is a de-facto standard in developing measurement applications that allows users to develop and monitor the behavior of their applications by dragging and dropping these icons and connecting them through wires in a block diagram that has a correspondent front panel window (GUI of the VIs), which acts either as a data input or a data output. LabVIEW is based on a graphical dataflow general-purpose programming language (known as G). G provides multithreading and multitasking, which presents major benefits for programming multicore processors and other parallel hardware such as FPGAs. LabVIEW is scalable across multiple OSs such as Windows, Mac, and Linux. It offers unrivalled integration with thousands of hardware devices and provides hundreds of built-in libraries for advanced analysis and data visualization. To save time and extend functionality, NI provides more than 100 add-ons of toolkits and modules for LabVIEW, in addition to hundreds of add-on tools developed by third-party partners. LabVIEW comes with hundreds of ready-to-use example VIs. Moreover, it integrates configurable Express VIs that encompass the most common functions out of more than 850 built-in signal processing, analysis, and mathematics functions that simplify development for a broad variety of applications, that LabVIEW offers, which significantly reduce the development time and complexity. LabVIEW allows access to more than 8,000 program examples submitted by fellow developers and NI engineers. It offers an extensive support for accessing measurement and test hardware. In addition to NI drivers, more than 9,000 free drivers for instruments from more than 350 third-party vendors are available online for free download through the Instrument Driver Network. In the rare event that a LabVIEW driver doesn't already exist, user also can import drivers from other programming languages to implement his/her own driver. LabVIEW supports standard APIs such as IVI and VISA to provide interface-independent communication with different platforms such as GPIB, PXI, VXI, USB, LXI, and others. In addition to the above mentioned features and capabilities, the common features and popular tools for remote laboratories development are:

- **Integration with external software:** *LabVIEW Application Builder* can compile VIs as C/C++ libraries for Windows, Mac, and Linux OSs. Likewise, LabVIEW can call external libraries via the *Call Library Function Node*. VIs can also be compiled as RESTful Web services that can be invoked via requests from clients using standard HTTP. Likewise, LabVIEW can import external Web services. LabVIEW provides a built-in Web server for publishing the Web pages (i.e., known as *Web Publishing Tool* or *Remote Front Panel*) that allows controlling its front panel directly from a Web browser. The embedded dynamic objects of VIs requires the *LabVIEW Run-Time Engine* (i.e., available for download at the NI Web site) to be installed, which enables the remote panel to load on the client's Web browser. If another client is controlling the VI at the same time, user is queued; if this is the case, the client operating the VI maintains its control only for a fixed time determined by the

Web server developer. A dialog box of VI server configuration allows the developer to enable the server and set its properties such as port number and TCP/IP access privileges needed for security.

- **Programming embedded devices:** *NI LabVIEW C Generator* allows generating ANSI-C code from LabVIEW VIs and porting it to any microprocessor (i.e., 8-, 16, or 32-bit). For FPGA programming, *LabVIEW FPGA Module* allows programming NI reconfigurable I/O (RIO) hardware targets such as CompactRIO, Single-Board RIO, and FlexRIO through graphical block-diagrams combined with ready-made FPGA I/O templates instead of using HDL. It also allows importing external VHDL code.
- **Programming PLCs:** *NI OPC Servers* provides support for OPC classic and OPC UA protocols and enables LabVIEW to communicate with many different PLCs and third-party devices and to develop HMI and SCADA applications.
- **Database connectivity:** *NI LabVIEW Database Connectivity Toolkit* enables communication with local and remote databases using LabVIEW programming instead of SQL. Alternately, complete SQL capabilities are also supported for advanced database functionality and flexibility. The toolkit works with any provider that adheres to the Microsoft ActiveX Data Object (ADO) standard or complies with the ODBC or Object Linking and Embedding Database (OLEDB). Given the diver DLL of a database, LabVIEW can call this DLL and gain access to the database using the Calling Library Function Node. Another approach is to utilize ActiveX function of LabVIEW to call Microsoft ActiveX Data Objects (ADO) and communicate with the database using SQL [91].

LabVIEW has an outstanding graphical programming capability and it is regarded as powerful development environment for data acquisition and tuning, though, it is week at numerical computing and processing and algorithm design. MATLAB, on the other hand, has powerful data processing and high-level technical computing capabilities. MATLAB is ideal for developing complex control algorithm (e.g., fuzzy logic, data processing, etc.) and simulation. Some remote laboratory applications combine the merits of both integrating them in the same application. In [92, 93] for instance, MATLAB is used for designing control strategy, while LabVIEW is used for data acquisition and GUI. Fortunately, LabVIEW allows hybrid programming and integrating MATLAB components into its environment—though the opposite is not possible—in four possible ways:

1) LabVIEW can act as an ActiveX client and imports a registered MATLAB ActiveX objects into an ActiveX container. In this case, MATLAB should be running as an ActiveX server.
2) MATLAB code can be compiled by the *MATLAB Builder NE* into a COM component to be registered and called by the LabVIEW *ActiveX Automation* with the option of programming its properties and methods.
3) MATLAB code can be compiled by the *MATLAB Compiler* into a C/C++ library to be called by the LabVIEW *Library Function Node*.
4) MATLAB m-file script can be embedded into LabVIEW applications using the *MathScript RT Module*, which connects the text-based I/O variables with the inputs and outputs of LabVIEW. *MathScript RT Module* offers math-oriented textual programming through that provides a native compiler for .m files and includes more than 750 built-in functions that are commonly used for math, signal processing, analysis, and control.

Summing up, LabVIEW is inherently suited for remote control applications with a strong support to a vast range of hardware, drivers, and industrial standards. It also provides a large set of software connectivity and communication tools. This is in addition to its seamless graphical programming and its Web publishing tools, which allowed users to deploy their applications in Internet in a very easy and straightforward step. Finally, its support for integrating MATLAB components complements it and converts it into an authentic IDE. All these reasons have contributed in the promulgation of LabVIEW and in making it the first choice among developers and the de facto platform for laboratory server software development. Thus, LabVIEW is adopted as a strongly recommended technology for laboratory server software development in LaaS architecture. Figure 47 provides a block diagram of the six main components of LabVIEW for remote laboratory development.

5.7.7. User Interface (UI)

Developer or users are free to develop their own UI with their preferred programming language and technologies upon the provided service description file. The final application built by users could be implemented either by embedding it into a client or an educational system or by desktop sharing techniques [94].

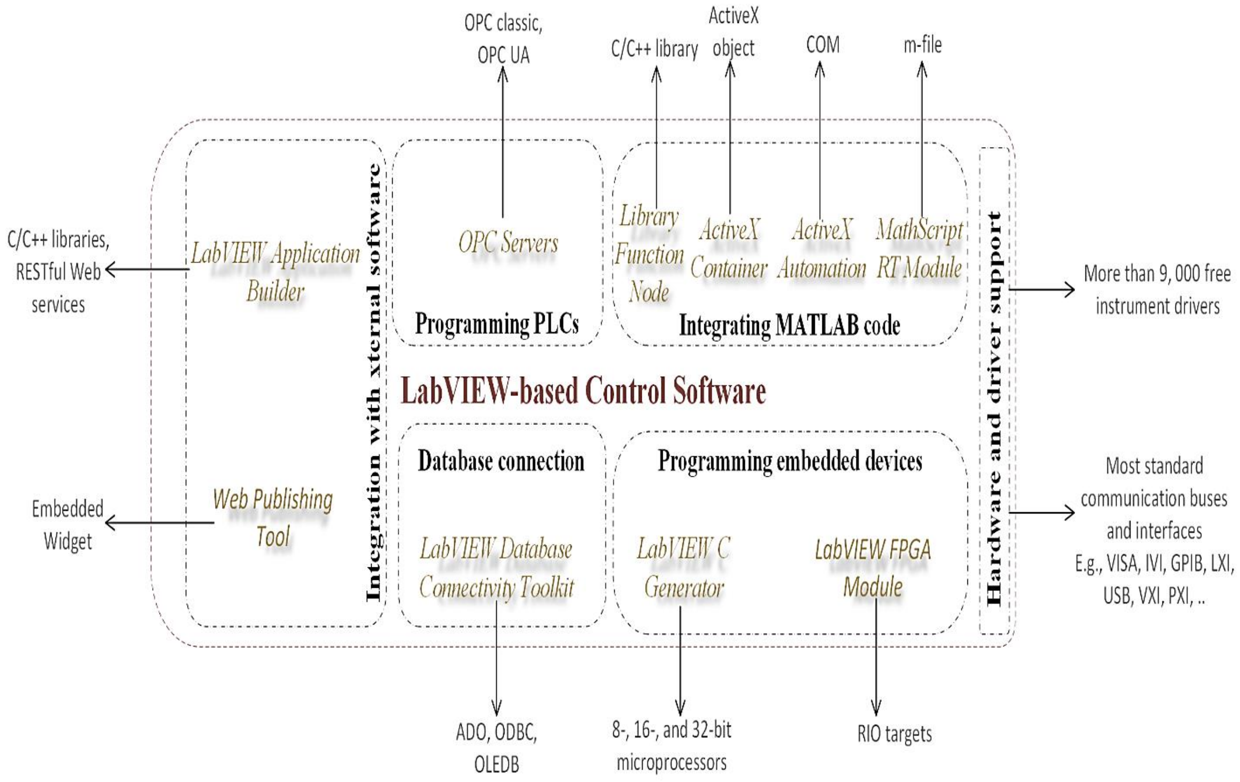

OPC classic,
OPC UA
ActiveX object
C/C++ library
COM
m-file
C/C++ libraries,
RESTful Web services
LabVIEW Application Builder
Integration with xternal software
OPC Servers
Programming PLCs
Library Function Node
ActiveX Container
ActiveX Automation
MathScript RT Module
Integrating MATLAB code
Hardware and driver support
More than 9, 000 free instrument drivers
LabVIEW-based Control Software
Database connection
Programming embedded devices
Embedded Widget
Web Publishing Tool
LabVIEW Database Connectivity Toolkit
LabVIEW C Generator
LabVIEW FPGA Module
Most standard communication buses and interfaces
E.g., VISA, IVI, GPIB, LXI, USB, VXI, PXI, ..
ADO, ODBC, OLEDB
8-, 16-, and 32-bit microprocessors
RIO targets

Figure 47. LabVIEW's main components for remote laboratories development.

Desktop sharing solutions have the advantage of providing direct access to the laboratory server software without further installation or external software layers and they might enable concurrent access. When the user logs on, he/she can access to the windows desktop with the maximum restrictions, he/she can only start the programs associated to the control of the workstation but he/she can't modify the configuration, shutdowns, or viewing the folders in its hard disk. This solution is requires excessive bandwidth to support continuous communication of screen updates and lacks security. Some features are offered for protection including data encryption and password security.

An extra security layer with stronger encryption could be added by tunneling over Secure Shell (SSH) or VPN connection. For remote laboratories applications, the user access must be controlled by a system hour reservation to support the simultaneous remote access of students. The typically adopted desktop sharing solutions for remote laboratories are: (1) Virtual Network Computing (VNC): it is a cross-platform protocol that is based on image transferring which slow its speed performance. It allows multi-connection but within the same session, so all users see the same thing; and (2) RDP: it is developed by Microsoft and only runs on Windows OSs. It is based on data transferring, which increase its speed performance rather than image transferring. RDP treats each connection as a new session but it is limited to a single connection. File transfer between the local and the remote controlled machines is not supports as in other desktop sharing software. In contrast to the objectives of the LaaS paradigm, desktop sharing solutions cannot be delivered, integrated, and combined as a loosely-coupled service and thus, they should be avoided.

External software applications provides GUI to the users by supporting interactive multimedia. Interactive multimedia can be developed by either plugin-based approaches—such as ActiveX controls, Java applets, and Rich Internet Applications (RIAs)—or Web standards-based approaches relying on native Web technologies such as JavaScript and AJAX.

ActiveX controls officially operate only with Microsoft's Internet Explorer and Microsoft Windows OSs. Programmers can write ActiveX controls in any language that supports COM, including C/C++, .NET, Java, and LabVIEW. ActiveX controls allows reusing existing software components inside Web browser or inside other software application. ActiveX controls are Microsoft Win32 components; they have full access to the operating system, with all the risk that this implies. Microsoft has implemented a registry of digital signatures to verify ActiveX controls. Signed ActiveX controls are embedded with an encrypted digital signature from the software publisher. This signature can be used to verify the source and integrity of the code it is attached to. Regardless of the presence of a signature, users must be sure they know what actions or features the control is intended to perform, and that they have a degree of trust in the purported publisher before allowing the control to run on their system.

Java applets are written using the Java programming language and translated into byte code, which is platform and Web browser-independent. The byte code can be loaded and compiled at runtime by the clients and then executed using JVM. The JVM will need to restart each time the browser starts afresh. Some applets require a later version of Java Runtime Environment (JRE), which will force the client to wait for a large download. The Java programming language and tools were designed with security in mind. Each Java application or applet can be given customized permissions to control what actions it is allowed to perform. Unsigned applets run within their own protected area in memory, which is known as a sandbox. The sandbox model prevents applets from interfering with each other or other processes on the system. Thus, security restrictions may make it difficult or even impossible for an untrusted applet to achieve the desired goals. Users or the JVM can grant signed applets permission to operate beyond the bounds of the sandbox. In addition to the sandbox concept, Java includes robust security libraries that provide encryption, certificate implementation, code signing, permissions, and authentication.

Java Applets and ActiveX controls, are similar technologies that are based on a downloadable code from the Web server to be executed on the local computer in order to provide a rich dynamic content functionality to the browser and to add interactivity to Web pages either by embedding the contents or running them on a separate window. However, both use different approaches to provide a similar functionality. Java Applets and ActiveX run on client and hence, do not overload the server. Java Applets contain 8-bit byte code, while ActiveX components contain full 32-bit native code. This is one more reason why Java is significantly slower than ActiveX components. ActiveX components are only downloaded the first time they are accessed and each time a new version of the software is updated. On the other hand, Java applets are downloaded every time they are accessed, which cause a potential load on the network.

Java Applets and ActiveX controls are heavily used in remote laboratories development for delivering interactivity and embedding objects into <*object*> elements. The LabVIEW's *Remote Frontal Panel* relies on ActiveX and it is by far the most prevalent approach for bringing laboratories online. Furthermore, LabVIEW per se supports the ActiveX automation and container functionalities, which are extremely useful in building remote laboratories applications. In the same remote panel of a remote laboratory application, developers can embed video streaming, guides, or any other external elements using ActiveX technology. In ActiveX automation, LabVIEW acting as a client can launch any ActiveX enabled application (ActiveX server) that is registered in the local computer such as Microsoft Excel, Internet Explorer, or even a LabVIEW VI Server. This is done be creating a reference to the ActiveX server, using the *Automation Open Function*, and passing this reference to the *Property Node* or *Invoke Node* functions. Then the reference is closed to release the object resources. LabVIEW can fill the automation server role as well and allows client applications (e.g., Visual Basic, Visual C++, and Microsoft Excel applications) to access its functionality programmatically. The type library "*labview.tlb*" provides information about LabVIEW automation server's objects, methods, and properties. In ActiveX container, LabVIEW contains or embed two types of objects or components, from other software packages, that are registered in the local computer: controls and documents. A control is an ActiveX object that can be embedded and controlled graphically. Optionally, it has an automation interface that enables controlling it programmatically (i.e., it appears as an automation "*refnum*" terminal on the block diagram). Documents are objects that can be embed in a container by either directly linking to the source file (i.e., any changes to that document are reflected in the container application) or by creating a new instance allowing the document to become part of the LabVIEW application. If an object does not support an automation interface, the terminal will have an invalid "*refnum*" and hence cannot be used with the automation functions. The properties of ActiveX controls or documents are configured either by the *Property Browser*, the *Property Pages*, or the *Property Nodes*. LabVIEW allows ActiveX event registration using the object "*Refnum*" and *the Register Event Callback* function to specify and register the event. A *callback VI* that contains the code to handle the event from the ActiveX object is created afterwards.

The *AppletVIEW Toolkit* from Nacimiento Software Corporation (www.nacimiento.com) allowed generating Java applets from LabVIEW front panels and VIs [95]. The open source software, *Easy Java Simulation (EJS)* [96], is a tool for creating of interactive dynamic simulations without the requirement of high programming skills. Users only need to provide the most relevant core of the simulation algorithm and the EJS software automatically generates all the Java code needed to run the final application as a Java applet to be embedded in a HTML page or as a standalone Java application in .jar format. EJS environment includes: (1) a model, where users can declare different types of variables and write the differential equations that establish how these variables change in time or under user interaction in order to describe the system; and (2) a view that provides a set of graphical elements and components (e.g., panels, buttons, shapes, images, etc.) to build a GUI interface in a simple drag-and-drop way. Users set a link between the model and the view to build the final application. Ordinary Differential Equations (ODEs) in the model can be numerically solved (i.e., using methods such as Euler–Richardson, Runge–Kutta, Fehlberg, etc.) by a built-in editor or by a connection to MATLAB/Simulink. External libraries can be imported and likewise graphical elements and components. EJS was implemented in several remote laboratories applications [97, 98] to provide graphical features owing to its capability of generating Java applets from either LabVIEW or MATLAB applications. The former approach is achieved using a stand-alone LabVIEW application server, dubbed *Java-Internet-LabVIEW (JIL)*, which acts as middleware that publishes LabVIEW VIS's on the Internet providing a TCP/IP wrapping to their control loops. The server performs an automatic scan of all the VI's controls and indicators, initializes the network input port, and waits for an incoming connection from a remote Java clients. A Java library file is added to EJS to allow it to automatically connect to the VIs and retrieves the list of controls and indicators URLs published by the JIL server. Afterwards, the developer links the URLs to the variables declared in the EJS model instead of specifying the entire model [97]. The latter approach is similar to the former. A special built-in link of EJS is used to manipulate Simulink models. This link enables connecting EJS's variables to the block variables of the Simulink model without requiring any modification of it. This link uses the *JMatlink* library [99] and the engine library to control MATLAB from EJS applications. It is direct if both software are in the same computer and if MATLAB and the EJS application are located in different computers a stand-alone Java server application, dubbed *Java-Internet-MATLAB (JIM)*, is used to allow an EJS application to use a remote MATLAB server. Similar to the local link case, the library *JMatlink* is used to control MATLAB from JIM server [98]. Another solution (known as JACOB) for bridging

MATLAB and JAVA applications through COM technology but only under Windows operating system is described in [100]. Figure 48 illustrates the role of ActiveX and Java applets in the development of remote laboratories under three typical scenarios.

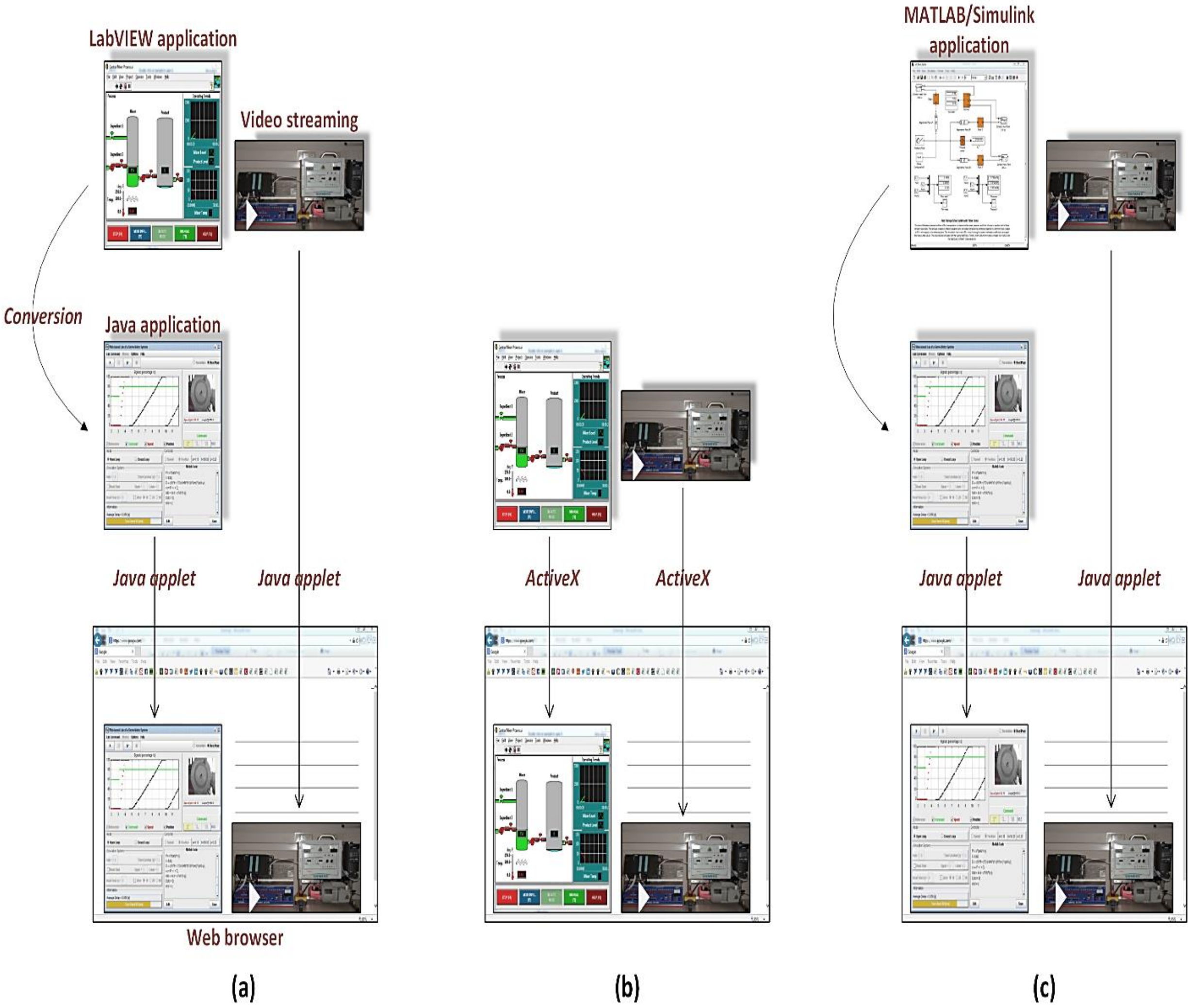


Figure 48. Three typical scenarios illustrating the role of ActiveX and Java applets in the development of remote laboratories.

In the first scenario, Figure 48(a), a LabVIEW application is converted into a Java application by a middleware technology (e.g., JIL, AppletVIEW Toolkit, etc.) and then this Java application along with other elements such as Webcam video streaming are embedded into the end user Web page as Java applets. In the second scenario, Figure 48(b), a LabVIEW application and other elements such as Webcam video streaming are embedded into the end user Web page as ActiveX objects. In the third scenario, Figure 48(C), a MATLAB/Simulink application is converted into a Java application by a middleware technology (e.g., EJS, JIM, JACOB, etc.) and then his Java application along with other elements such as Webcam video streaming are embedded into the end user Web page as Java applets.

RIAs integrate multimedia, graphics, animations and interactivity into webpages, features that are not found in the above mentioned solutions. They are plug-in-based and once the plugin is downloaded, it does not need to be downloaded every time the page is displayed. This reduces application load time, bandwidth requirements, and server load. For security purposes, RIAs run their client portions within a special isolated area of the client desktop called a sandbox. The sandbox limits the visibility and access to the local machine. In remote laboratories, Adobe Flash is the commonly used for building RIA. A famous example is the VISIR remote laboratory discussed in Chapter 3. In VISIR, a GUI developed Flash communicates with a LabVIEW application though an XML based protocol. Adobe Flash is more compatible with all OSs and Web browsers. Other competitors RIAs are Microsoft Silverlight and Oracle JavaFX. However, these are recent technologies with no significant adoption in remote laboratories applications yet.

Recently, several approaches endeavored to shift from plug-in solutions towards native Web technologies such as AJAX, and JavaScript [72, 101]. AJAX is a programming technique to build rich and dynamic web applications that quickly respond to user requests. AJAX enables JavaScript to communicate directly with the Web server using the *XMLHttpRequest* object, which requests and updates only the required data on the Web page instead of reloading the entire Web page. AJAX uses standard technologies that all Web browsers already support, except *XMLHttpRequest*, such as HTTP, JavaScript, XHTML, XML, Document Object Model (DOM), and CSS to request, retrieve, convert, and present the data on the Web page. All recent Internet browsers and platforms support Ajax and there is no need to install any software on the client machine. AJAX per se does not provide the rich multimedia and graphical capabilities provided by RIAs. Both solutions usually co-exist and there is no estimation that AJAX would replace RIAs in the near future. Nevertheless, embracing Web native technologies will be a priority in order to achieve maximum compatibility and mobile access as discussed in Section 5.4.2. Another approach towards open standards is found in [56], where OpenSocial widgets are built using JavaScript and DHTML in order to render in any widget engine or container.

5.8. Introductory Examples

After describing in details the proposed LaaS paradigm including all its stages and components. The idea can be briefly resumed in the following two demonstrative examples.

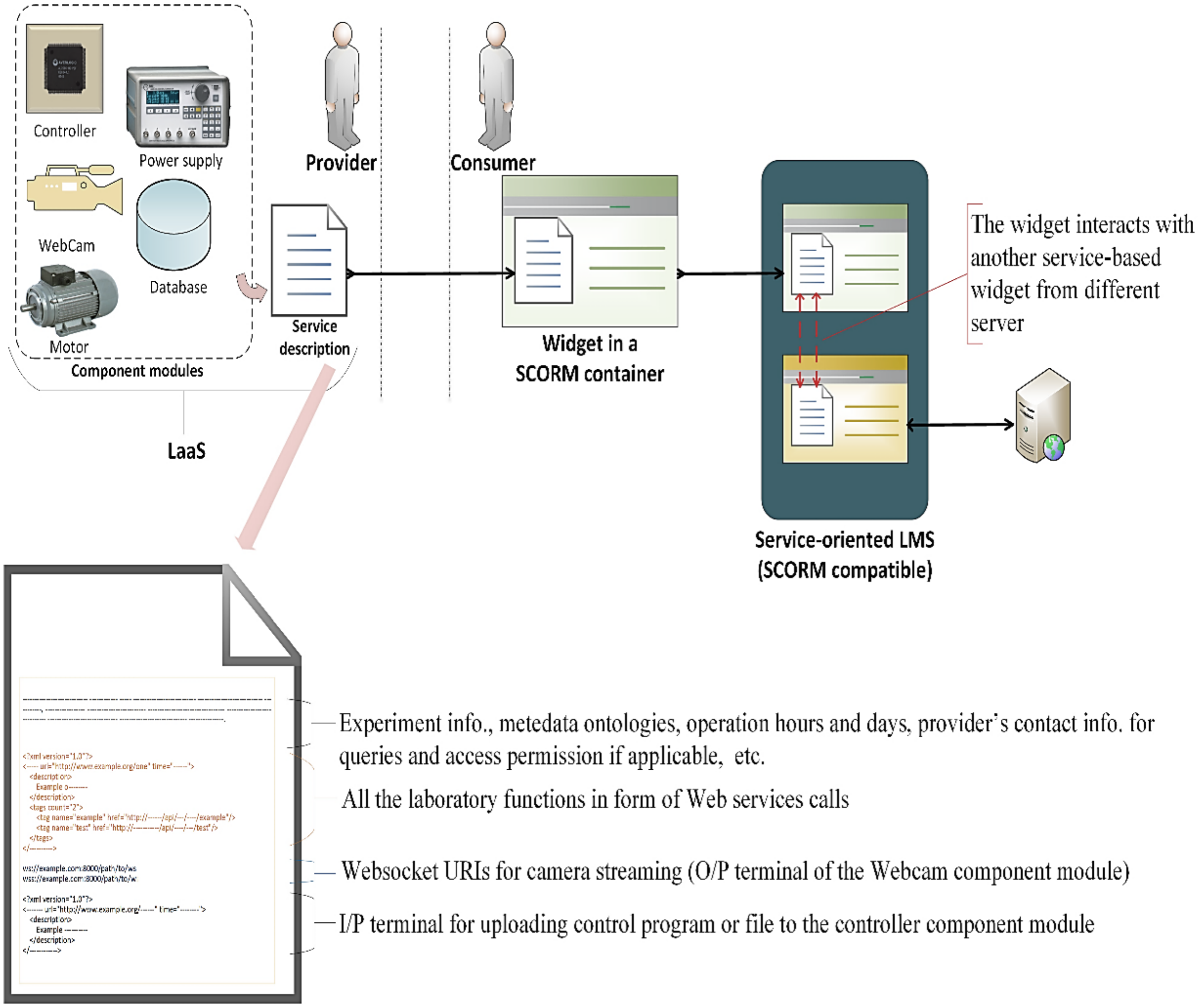
5.8.1. Example 1

Consider the following scenario shown in Figure 49, where the owner of the modular laboratory provides it as a set of services, according to the LaaS paradigm, to the consumer.

In this example, the laboratory provides an experiment for implementing a control strategy on an electric motor. User uploads his/her PID control program to the controller, changes the PID parameters (e.g., speed and position), and monitors the feedback effect of different control loops. The laboratory has 5 main component modules: a motor, a controller, a Webcam, a power supply, and a database. All the provided services of the laboratory (i.e., the functions, the I/O terminals, and the I/O connectors) are listed in the service description file (e.g., a pdf file) to be read and consumed by the consumer. The service description file is divided into two main parts:

1) The first part contains information such as:
    - A description about the experiment, how it works, objectives, etc.
    - Metadata ontologies to help the mediator or the service broker to list it in the marketplace.
    - Days and hours in which the laboratory is (or not) available if applicable.
    - Access policy and contact or query forms for applying access if applicable.

2) The second part contains all the laboratory functions, I/O terminals, and I/O connectors in form of either Web service or URI for streaming over low level protocol or Websocket. The Webcam component module will provide an output terminal for video streaming. The controller component module will provide an input terminal to allow user to submit his/her own program code file.



The widget interacts with another service-based widget from different server

```
<?xml version="1.0"?>
<----- url="http://www.example.org/one" time="------">
   <descript on>
      Example o--------
   </description>
   <tags count="2">
      <tag name="example" href="http://------/api/---/----/example"/>
      <tag name="test" href="http://----------/api/----/---/test"/>
   </tags>
</---------->

ws://example.com:8000/path/to/ws
wss://example.com:8000/path/to/w

<?xml version="1.0"?>
<------- url="http://www.example.org/-----" time="-------">
   <description>
      Example -------
   </description>
</---------->
```

Experiment info., metedata ontologies, operation hours and days, provider's contact info. for queries and access permission if applicable, etc.

All the laboratory functions in form of Web services calls

Websocket URIs for camera streaming (O/P terminal of the Webcam component module)

I/P terminal for uploading control program or file to the controller component module

Figure 49. LaaS scenario of Example 1.

The service broker (or the provider him/ herself) indexes the LaaS in the market place (i.e., not shown in the figure for simplicity). The consumer (or user) allocates the laboratory in the marketplace and downloads the service description file. Using the file, the consumer builds his own widget application and wraps it in a SCORM container. User, afterwards, will be able to load his SCORM package in any SCORM-compatible LMS. User might not consume all the provided services in the same widget, instead he/she might link some of the provided services to other service-based object. For instance, user might want to introduce the PID parameters though his own widget but monitor the results in an external widget from different server that are specialized in building charts with regard to the received parameters. Using a service-oriented LMS, user will be able to use a SCORM package installed at other LMS through his own LMS (or his institution's LMS).

5.8.2. Example 2

Consider the same scenario of the previous example but with some modifications as shown in Figure 50, where the provider doesn't wish to share some of his laboratory component modules, the database and the power supply.

In this case, the provider tries as possible to reduce the load on his/her own equipment and facilities and instead leaves it to the consumer to connect his/her own component modules as long as they adhere to the same adopted standard. To accomplish this, the provider instead provides the consumer with standard-based I/O connectors to connect his/her component modules. In this example, the consumer connects his/her database using an ODBC connector and connects his/her power supply using IVI/VISA connector. The rest of the procedures are same as the previous example.

This aim of this example was to explain the concept of modular components and interchangeability using two components, however the idea is much broader and can be applicable to any other laboratory component.

5.9. Summary

This shift from Web 2.0 to Web 3.0, and consequently from eLearning 2.0 to eLearning 3.0, has demanded a rethinking in the way leaning objects including remote laboratories are developed and delivered. Even though Web 3.0 is still purely hypothetical, it is anticipated that future Web will embrace: the semantic Web and artificial intelligence; the WoT and the omnipresence of everyday connected objects, and the mashup of loosely coupled services. These three speculated mainstreams give us insights about the characteristics of next generation online learning environments and accordingly the LaaS paradigm for developing and implementing modular remote laboratories suited for this kind of environments was developed.

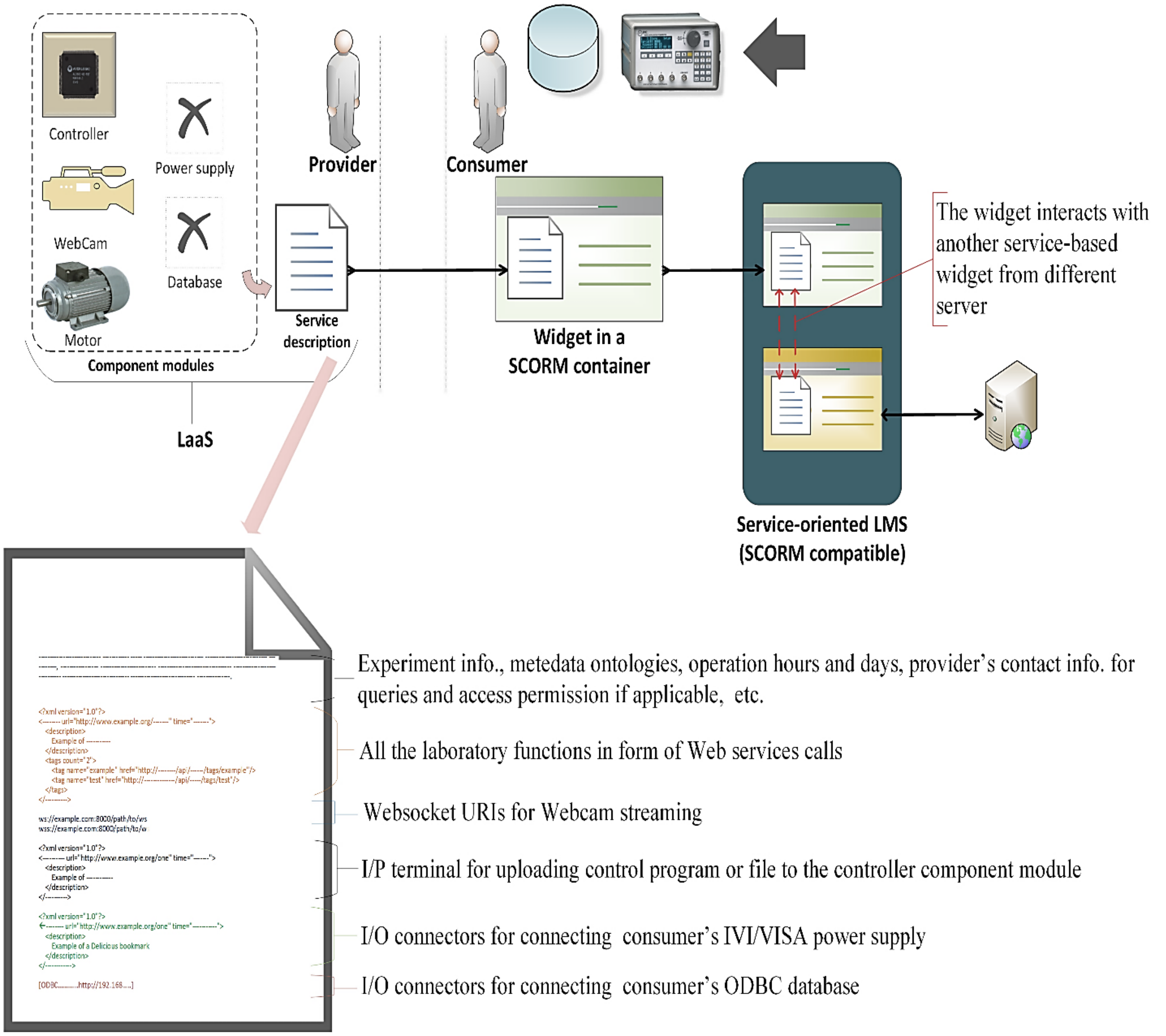

Controller
Power supply
WebCam
Database
Motor
Component modules
LaaS
Service description
Provider
Consumer
Widget in a SCORM container
Service-oriented LMS (SCORM compatible)
The widget interacts with another service-based widget from different server
Experiment info., metedata ontologies, operation hours and days, provider's contact info. for queries and access permission if applicable, etc.
All the laboratory functions in form of Web services calls
Websocket URIs for Webcam streaming
I/P terminal for uploading control program or file to the controller component module
I/O connectors for connecting consumer's IVI/VISA power supply
I/O connectors for connecting consumer's ODBC database

Figure 50. LaaS scenario of Example 2.

Modular remote laboratories are based on interchangeable component modules that expose their I/O terminals or their I/O connectors (i.e., if they physically don’t exist or unavailable) in an independent and an abstracted way. Some components can be modularized and some are fixed and cannot be modularized or interchanged programmatically (e.g., laboratory server and DAQ board). The idea beyond modularizing remote laboratories is to facilitate maintenance, reusability, and interchangeability of components seamlessly and programmatically. In this sense, if the I/O terminals and connectors of all the component modules of a remote laboratory are provided in a “service description file” in order to allow consumers to get clues on them, the consumer would be able to consume them separately. Furthermore, if one of the component modules is not available and the appropriate I/O connectors are provided instead, the consumer could replace this module with his/her own one instead of the fully-reliance on the provider’s equipment.

In order to facilitate interchangeability, it is recommended to rely on well-known standard connectors such as ODBC for database and IVI/VISA for instrument. Thus, if the ODBC database I/O connectors are provided to the consumer in the “service description file”, he/she would be able to connect his/her own ODBC-compliant database independently of its model or manufacturer as long as it adheres to the standard, and likewise for his/her IVI/VISA-complaint instruments.

LaaS builds upon the modular remote laboratory concept and implies the delivery of the entire laboratory functions and components in the “service description file” as a set of abstracted services. LaaS follows SOA and fulfills its essential requirements in terms of interoperability, service description, and exchanging messages. It defines the relation between laboratory providers (i.e., providers of the “service description files”), service broker repository (i.e., Web portal in which “service description files” are indexed), and laboratory consumers (i.e., who build an end-user application upon the provided services).

All the laboratory functions are implemented using—but not limited to—resource-oriented Web service, REST, owing to its simplicity and high performance, as well as, its homogeneity with Web applications in general and mashup applications in particular. Activity-oriented Web service, SOAP, is another alternative for advanced applications with high level QoS requirements. Other middleware technologies such as CORBA, .NET Remoting, RMI, and DDS were excluded—despite their higher performance—for any of these reasons: (1) firewall restrictions; (2) complexity; and (3) platform dependency. For server pushing and persistent connections like data streaming (e.g., Webcam video streaming or measurement streaming), encoding over low level protocols such as TCP and UDP is provided as a URI with public IP address instead of Web service. The LaaS paradigm can be resumed in the following demonstrative case study.

Consumer can build the end-user application using any technology or programming languages he/she prefers. Consumer can also build customizable applications suited for service-oriented LMSs and can couple and mash up the laboratory with other interoperable learning objects across the Web. LaaS is suited for service-oriented LMSs as it exposes all its functions using Web services and other low level protocols. For example, if a LaaS is wrapped (i.e., using some of its exposed services) into a SCORM package as an application container it could be easily imported in any compatible service-oriented LMS allowing the rest of the exposed services to communicate with external learning object within or outside the LMS. Native-Web technologies such as AJAX and HTML5 are recommended in order to facilitate mobile access through Web apps, but consumer can also use RIAs.

If the laboratory is made available, it should be accessible unless another session is currently running by another user. A mechanism for checking "whether it is currently available or not" should be included in its design (e.g., using a Web service call). Else, the consumer should contact the provider for enquiries. The consumer can also build his/her own scheduling system for a large scale deployment with numerous groups and students. Scheduling system is out of the scope of the LaaS paradigm as it focuses on the lowest level side in order to maintain the "service description file" as abstracted as possible.

This page intentionally left blank.

Chapter 6

6. Modular Remote Laboratory Prototype

6.1. Introduction

In this chapter, LaaS is going to be explained by a simple practical example as a proof-of-concept. In this example, a modular remote laboratory prototype is going to be developed and delivered following the LaaS paradigm. The given laboratory is a Motor-tacho laboratory and the idea of its experiment is very simple as it aims to emphasize the LaaS concept rather than delving into the technical details of the experiment. Despite its simplicity, it includes many features that are common in complex applications—in order to proof the concept reliably—such as hybrid programming using LabVIEW and MATLAB, database access, and video streaming.

In this experiment, user feeds the motor with a voltage rang from 0V to 5V and monitors the corresponding voltage value measured by the tachometer. A control strategy is implement so that if the applied voltage is greater than 5V it will be automatically modified and introduced as only 5V. Likewise, if the applied voltage is lower than 0V, it will be introduced as 0V. The tachometer measurement is streamed continuously until the user either stop the experiment or insert a new input voltage value. Each time the user inputs a value, it is automatically recorded and stored temporarily. Finally, when the user stops the experiment, all the inserted input voltage values—previously stored—are retrieved and listed in the database of the user (i.e., not the database of the provider). As well, the last measuring value of the tachometer is retrieved. A descriptive flowchart of the operation cycle is provided in Figure 51.

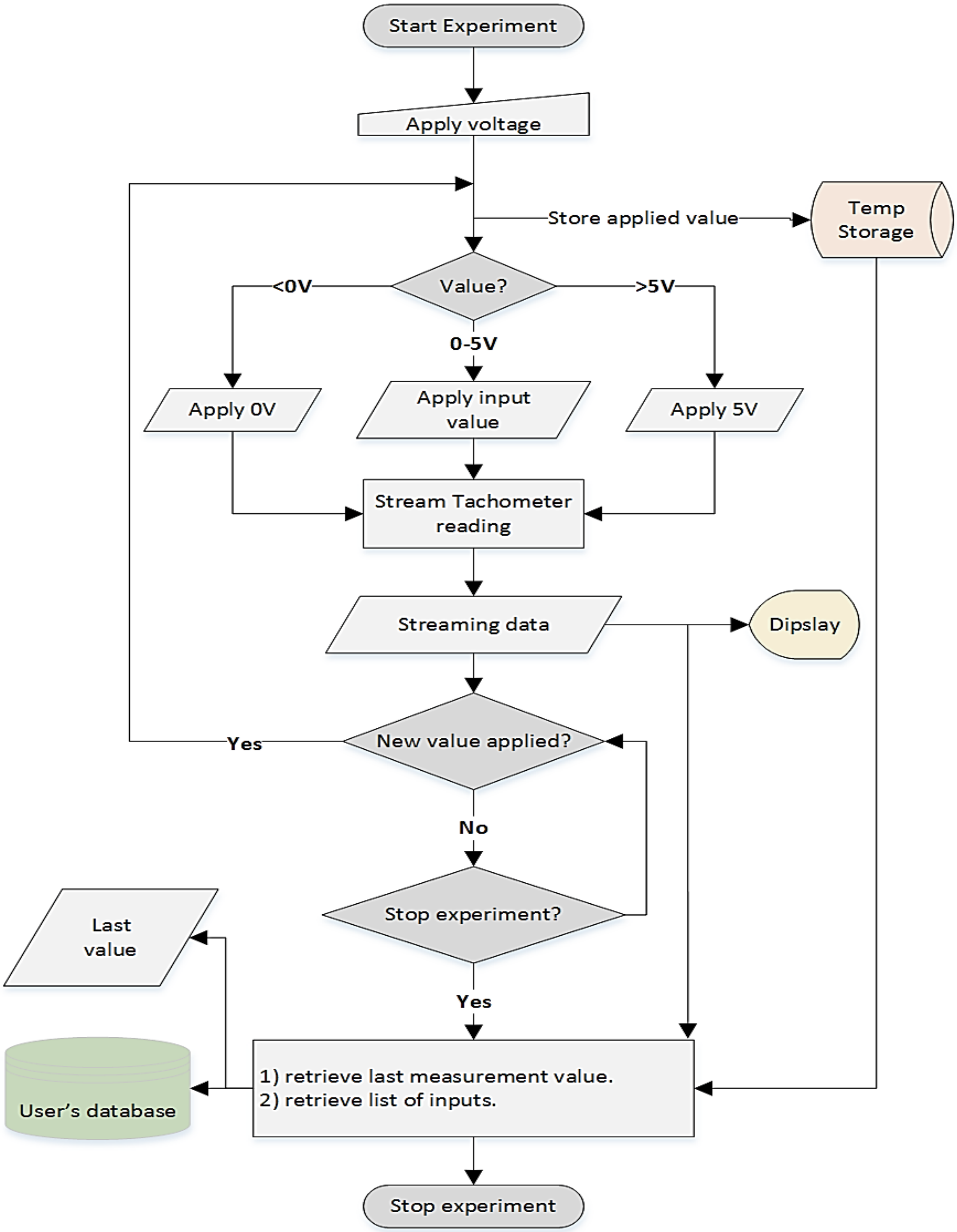


Figure 51. Flowchart of the motor-tacho laboratory's operation cycle.

The motor-tacho laboratory has three component modules and one of these modules, the database, adheres to a standard connector, ODBC. This laboratory requires that the consumer connects his/her own ODBC-compliant database to retrieve the list of input voltage values. The component modules are delivered following the LaaS paradigm as shown in Figure 52.

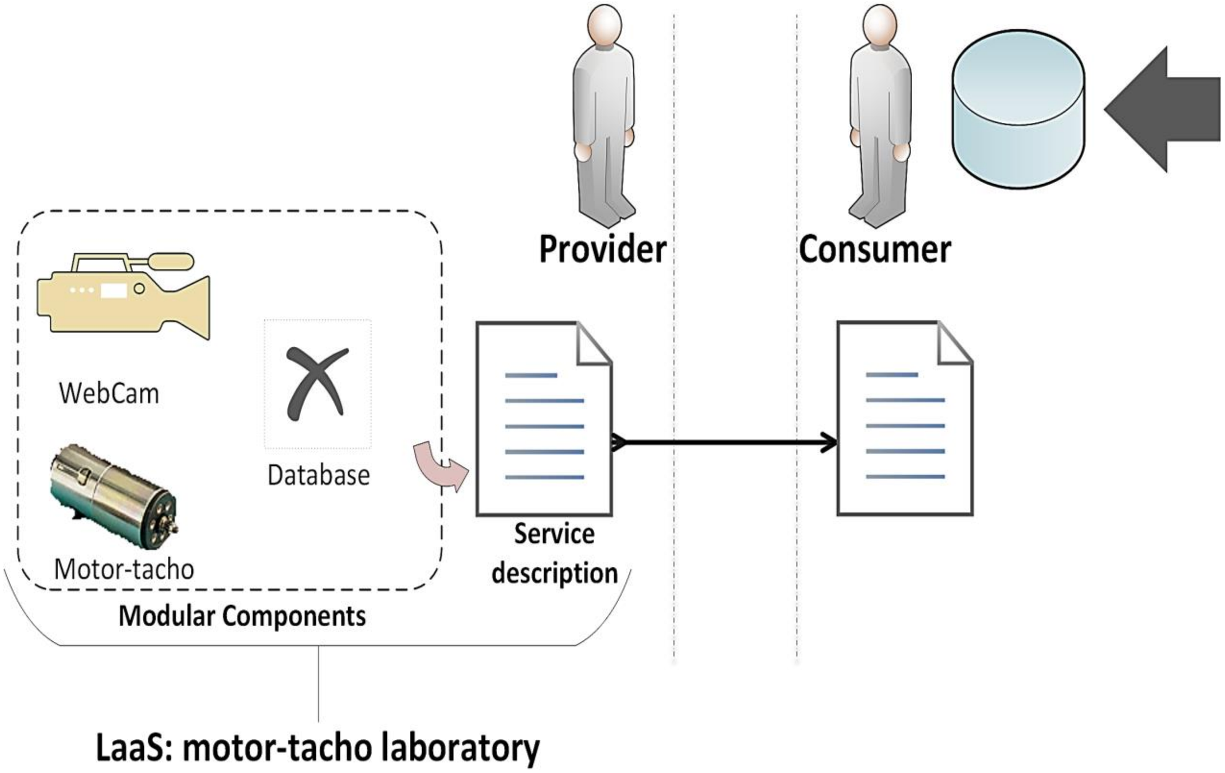


Figure 52. LaaS: motor-tacho laboratory.

The service description file will include: a description of the laboratory and the experiment; an ODBC connector for the database, Web services calls supported by the laboratory; video streaming URI; and tachometer measurement streaming over TCP URI. This was a brief introduction to the experiment and in the next subsections the laboratory is described in details.

6.2. Development

6.2.1. Hardware

The hardware components of the motor-tacho laboratory consists of a DAQ card, a motor-tacho, and an integrated Webcam as shown in Figure 53.

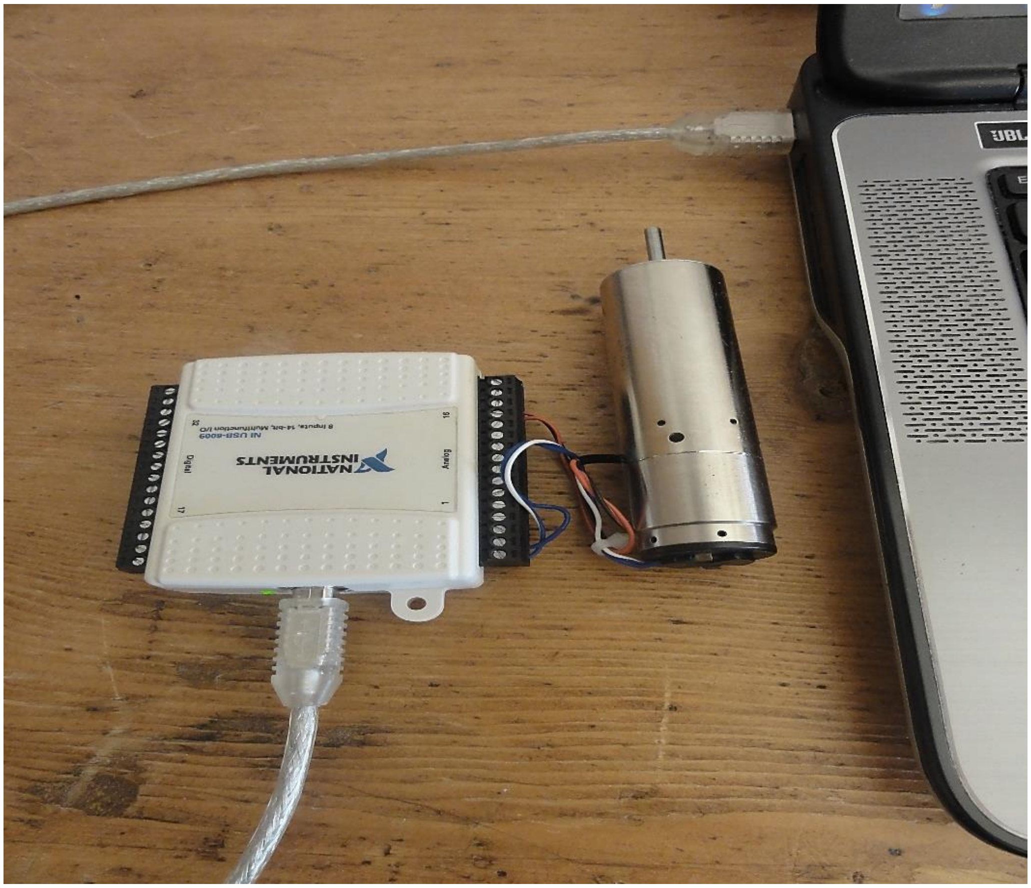


Figure 53. Motor-tacho laboratory.

The DAQ card is a commercial NI USB-6009 card from NI. It provides connection to 8 single-ended Analog Input (AI) channels (or 4 differential), 2 Analog Output (AO) channels, 12 Digital Input/Output (DIO) channels, and a 32-bit counter with a full-speed USB interface. The card pin-outs are shown in Figure 54.

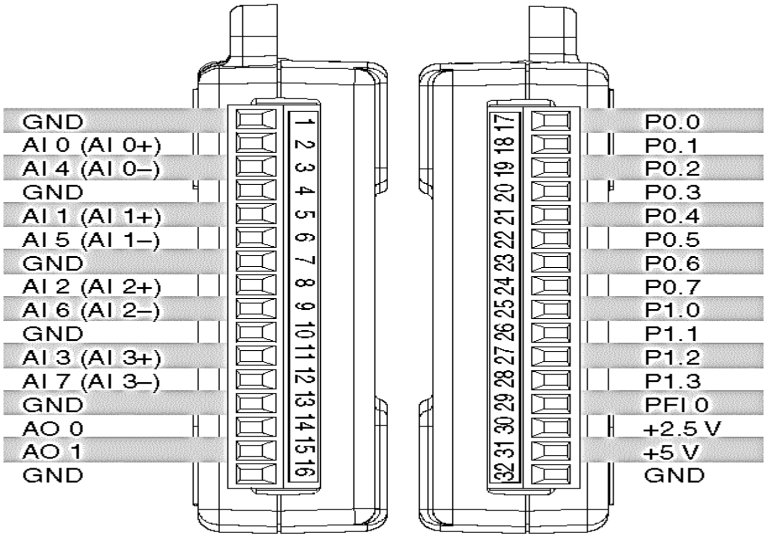


Figure 54. NI USB-6009 DAQ card pin-outs [102].

The card has an AI resolution of 13 bits single-ended (14 bits for differential); a maximum AI sample rate of 48 kS/s; a ±10 V single-ended AI range; an AO resolution of 12 bits; a maximum AO update rate of 150 Hz; a 0 to +5 V AO range; an AO impedance of 50 Ω; and an AO current drive of 5 mA [102].

The second hardware component is a commercial 28GD11 -222E/404E motor-tacho from Portescap (www.portescap.com). It has a typical starting voltage of 0.40V; a no-load speed of 5200 rpm and current of 28 mA; and a measuring voltage of 18V. The motor input is connected through the pins AO 0 & GND (14 & 13) of the DAQ card and the tachometer output is connected to the pins AI 0 & GND (2 & 1).

6.2.2. **Laboratory Server Software**

The laboratory server software was developed using LabVIEW. The DAQ card was first identified in the "Measurement & Automation Explorer" of LabVIEW then a new project was created to build the software. The "block diagram" of the developed software is shown in Figure 55.

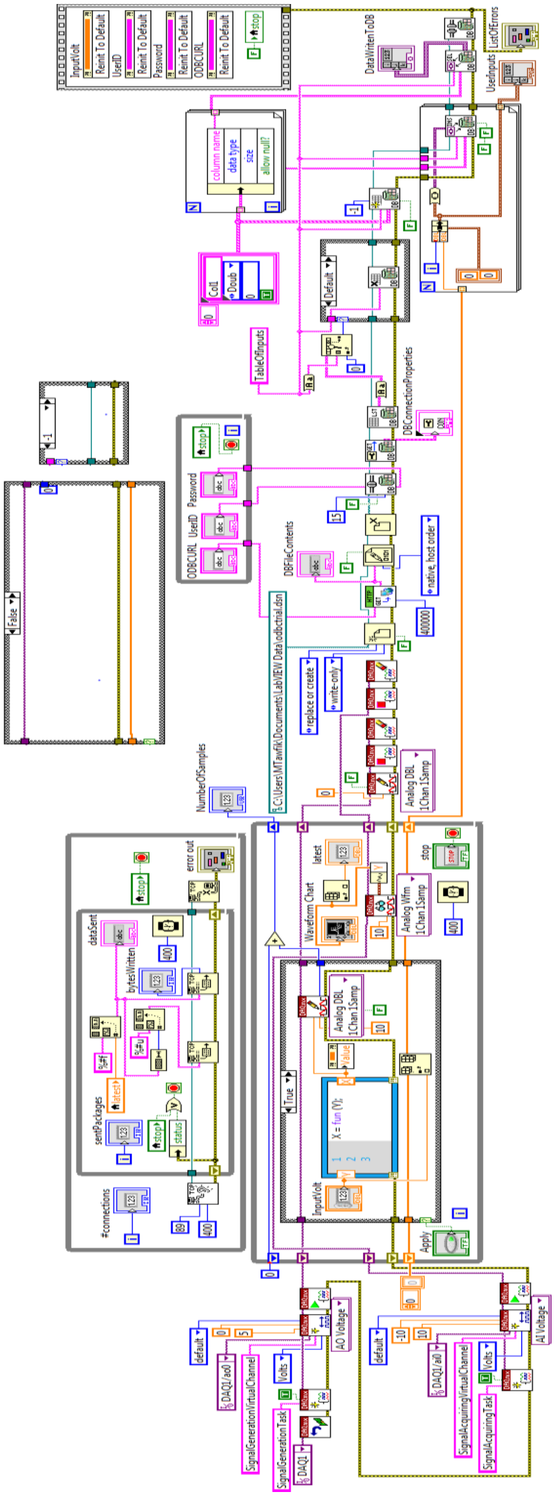
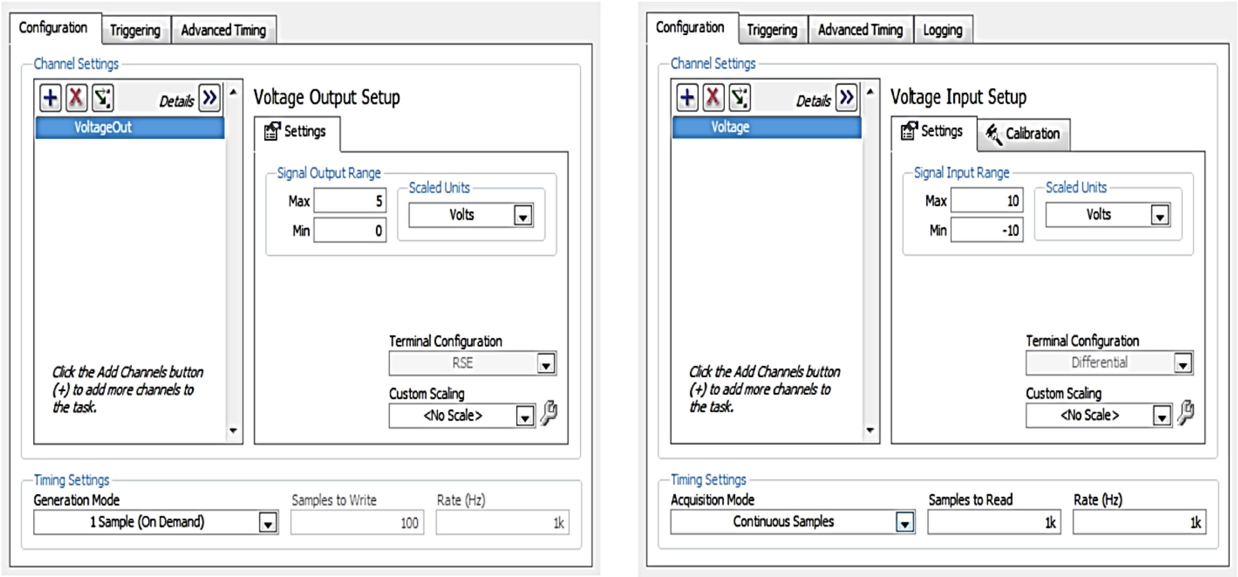


Figure 55. Block diagram of the motor-tacho laboratory's software.

The DAQ device is identified as “DAQ1”. The code first assures that the device is set to reset in order to avoid software crashing. Two DAQ tasks are created within the project for generating and acquiring signals and they are named “SignalGenerationTask” and “SignalAcquiringTask”, respectively. Likewise, a signal generation virtual channel is created for the “SignalGenerationTask” with a voltage range from 0 to 5 V and a signal acquiring virtual channel is created for the “SignalAcquiringTask” with a voltage reading range from -10 to 10 V. The configurations of the DAQ tasks are shown in Figure 56.



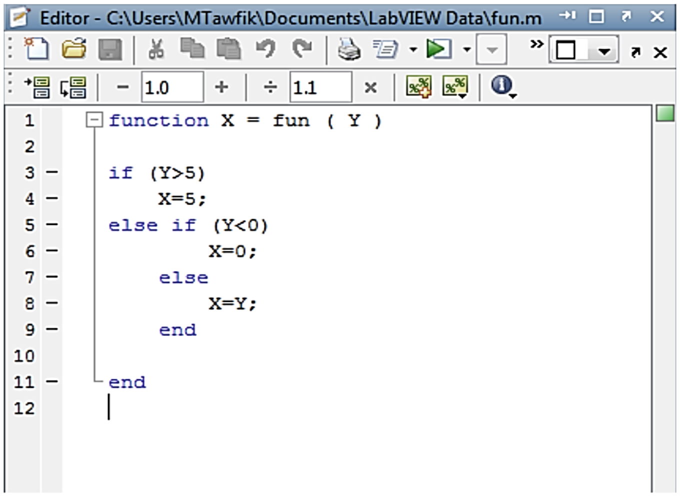
VoltageOut
Voltage Output Setup
Signal Output Range
Max 5
Min 0
Scaled Units: Volts
Terminal Configuration: RSE
Custom Scaling: <No Scale>
Timing Settings
Generation Mode: 1 Sample (On Demand)
Samples to Write: 100
Rate (Hz): 1k

Voltage
Voltage Input Setup
Signal Input Range
Max 10
Min -10
Scaled Units: Volts
Terminal Configuration: Differential
Custom Scaling: <No Scale>
Timing Settings
Acquisition Mode: Continuous Samples
Samples to Read: 1k
Rate (Hz): 1k

Figure 56. Configurations of the DAQ tasks.

Both DAQ tasks are connected to a “While Loop”, which is responsible for persistent reading and writing of data. When the user introduces a voltage value in the “InputVolt” numeric control and presses the “Apply” push button control, the “Case Structure” is set to “TRUE” instantaneously and its internal code is read—because the “Apply” push button operation is set to “Latch when released”. The internal code includes a “MathScript Node” for reading a control code written in MATLAB. This control code retrieves the input voltage value inserted by user from the “InputVolt” numeric control and outputs either: the same value if the input value lies in the range of 0-5V; 5V if the input value is greater than 5V; or 0V if the input value is less than 0V.

i.e., the MATLAB control code is written in an .m file named “fun.m” (i.e., the name of the file should be the same as the name of its function) as shown in Figure 57.



```
function X = fun ( Y )

if (Y>5)
    X=5;
else if (Y<0)
        X=0;
    else
        X=Y;
    end

end
```

Figure 57. MATLAB control code written in an .m file.

This file is imported to the “MathScript Node” and the input and output values (X & Y) are assigned accordingly. LabVIEW saves the file in the “LabVIEW Data” folder and links it to the “Dependencies” of the project.

The output value of the “MathScript Node” (X) is attached to a “Property Node” of the “InputVolt” push button control in order to change automatically its value if it goes over or below the range of 0-5V. The input value is sent to the “DAQmx Write.vi” of the signal generation task in order to feed the motor. The “DAQmx Write.vi” is configured as “Analog DBL 1Chan 1Samp”. Just after the motor starts to rotate, the “DAQmx Read.vi” of the signal acquiring task retrieves the measurements from the tachometer. The “DAQmx Read.vi” is configured to “Analog Wfm 1Chan 1Samp”. The signal is acquired as a waveform and then decomposed and its data value is sent to the “Waveform Chart” indicator to be viewed by the user. The data is also indexed in an array in order to allow reading its latest values for streaming through the “latest” numeric indicator.

All the input voltage values introduced by the user are inserted in an array and this array is shift registered during the “”While Loop” iterations in order to be used after the end of the “While Loop” execution. Similarly, the number of samples written per channel by the “DAQmx Write.vi” are shift registered for final reading after the end of the “While Loop” execution. ”Wait (ms)” timing is set to 400 for the “While Loop” to facilitate observing values.

i.e., Since the tachometer readings yields a continuous waveform, it is not possible to stream this wave form using Web service calls, and therefore, a code is developed for TCP streaming of the tachometer’s output during the execution of the “While Loop” of the read/write tasks. Two “While Loops” are created for streaming as shown in Figure 43. One for listening to incoming connections over the TCP port 89 and for calculating the number of connections. The other one is created for streaming the data if a connection is realized and for calculating the sent packages, which is approximately equal to the number of readings of the “DAQmx Read.vi”. That is why the “Wait (ms)” timing of both loops are equal and are set to 400. The data is retrieved from the “latest” numeric indicator (as a local variable). This input data (i.e., Double type data) is converted to a String through the “Format Value” function. The introduced format syntax “%#f” removes trailing zeros and returns a floating-point number with fractional format. Two “TCP Write” functions are required. One for sending to the consumer the length of each package prior sending it to him/her. The format syntax “%#u” cast the package length to an unsigned decimal integer number of bytes. The other “TCP Write” function is used to send the package data. This sequence should be clearly described to the consumer to allow him/her receiving the readings correctly. If any error occurred during the TCP connections the entire system automatically stops as the “stop” push button control of the main task read/write “While Loop” is shared by the “while Loops” through “local variable” copies and once the system stops it is reverted back to its “TRUE” value for the next session.

So, once the user finishes the experiment and presses the "stop" push button control, the acquiring and generating signal tasks are stopped and cleared. As well, the streaming loops are stopped. The next action is to connect to the database of the user (not the provider) through ODBC connector and create two columns one for numbering (e.g., 0, 1, 2, etc.) and the other for all the introduced voltage inputs by the user and finally close the database connection.

In order to connect to the user's database, first a Data Source Name (.dsn) file is created (or replaced if initially found) and stored at any path at the provider's server in order to be made available by the laboratory server software code for accessing it and get information about the remote database (the user's database). In this case the file path is "*C:\Users\MTawfik\Documents\LabVIEW Data\odbctrial.dsn*". The user transfers his/her own .dsn file through HTTP URL and the software copies the content of the user's file (which contains information about the user's database) to the local created "odbctrial.dsn" to be accessed by the laboratory server software. The content of the .dsn file includes the name of the database, the username, and the address of the remote server hosting the database. If necessary, the user is required to enter his/her database credentials (i.e., username and password).

Now that the laboratory server software has clues on the user's database through ODBC connector, it starts to communicate with this database through the "LabVIEW Database Connectivity Toolkit". The software opens a connection with the database using the information provided by the .dsn file and starts to search for the table "TableOfInputs" and drops it if found as it is going to create a new clean one with two columns of Double data type, col1 and col2. Once the two columns are created, the software starts to insert the user input values retrieved after the end of the execution of the read/write tasks "While Loop" into the created columns. This is done by the "*DB Tools Insert Data.vi*". The array of columns is unbundled in order to extract all the names of the columns using a "For Loop" and then pass them to the "*DB Tools Insert Data.vi*". The retrieved array of user input values is bundled with another array for numbering (i.e, 0, 1, 2, etc.) in order to create a cluster of data to be inserted in the two columns. The iteration is also realized by a "For Loop" and the numbering array is created by the iteration (i) terminal of the loop. The number of iteration of the loop is determined by the number of elements indexed after the end of the execution of the read/write tasks “While Loop” (i.e., the array of user inputs). After realizing this task, the data is written to the database and the communication is closed. Finally, the input values introduced by the user are removed and the controls are reinitialized to their default values. The front panel of the software is shown in Figure 58.

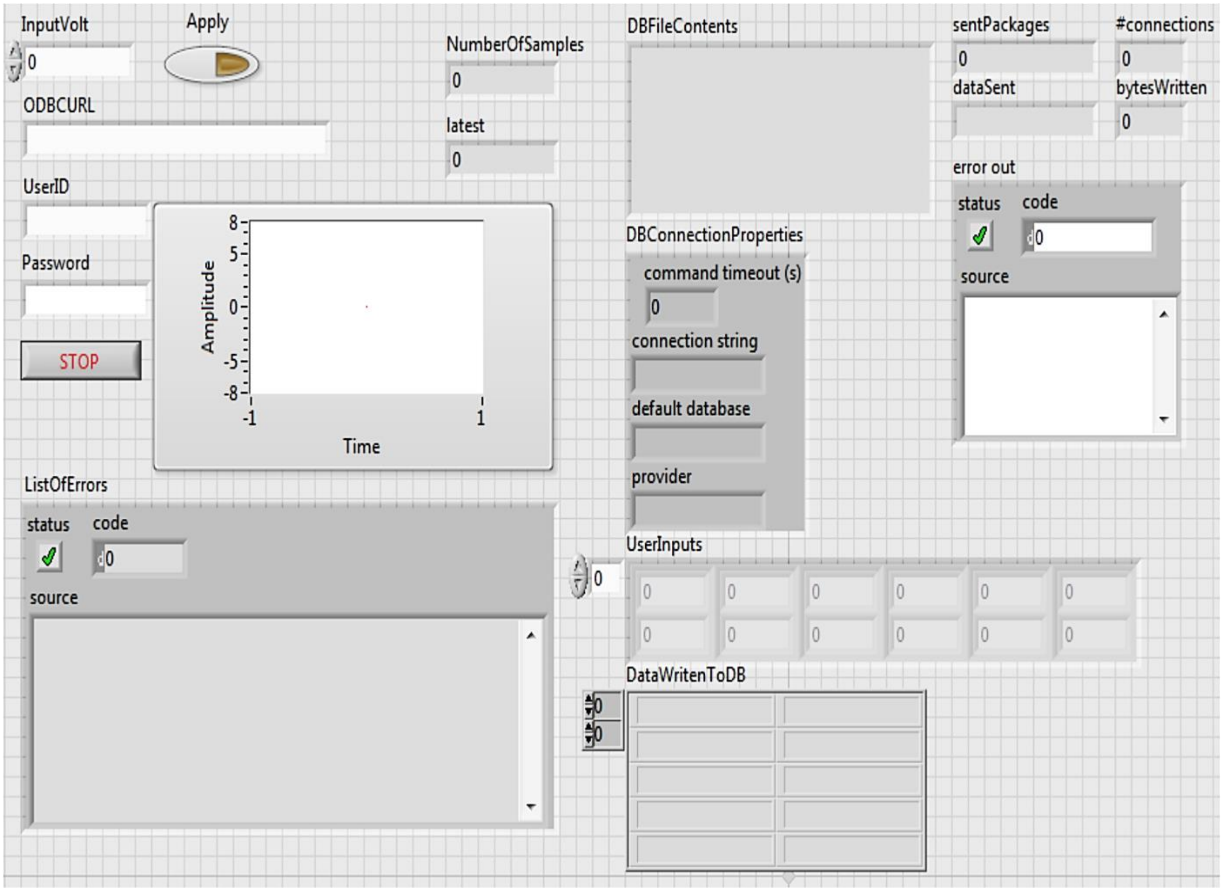


Figure 58. Front panel of the motor-tacho laboratory’s software.

The next step is to deploy the inputs and the outputs of this software application as Web services. Foremost, the LabVIEW *Application Web Server* should be running and enabled. In this example it is listening to the default port, 8000. *NI Application Web Server* shouldn’t be confused by the LabVIEW *System Web Server*, which by default listens to the port 3580. It shouldn’t also be confused with either the *VI server* or the *Remote Panel Server*, and different ports should be assigned to each of the aforementioned servers in order to avoid communication errors. The *LabVIEW Application Web Server* is configured from *Tools>Options>Web server>Configure Web Application Server* or through the Web browser at the address *http://localhost:3580/* by default. Through the configuration panel, the laboratory owner can decide who can access his/her exposed Web services by listing certain machine addresses or otherwise leave it open to the public. This option is very useful in case prior access agreement is required.

It is imperative to create a proxy VIs between the main application VI (i.e., the developed software shown in Figure 58) and the consumer because of two principle reasons. First, LabVIEW Web service tool creates a copy of the implemented VI in memory independently of the LabVIEW application and hosts the VI in the LabVIEW *Application Web Server*, which is in charge of dealing with incoming Web service requests. From the provider's perspective, it is important to visually monitoring the VI operation and its associated bugs. In this case the proxy VIs will be deployed and running on memory and dealing with the incoming Web service calls while the main software application will run in the same local machine but through the LabVIEW environment and hence visible to the provider or the laboratory administrator. The second and the most important reason is that Web service cannot run with Loops owing to the inherent HTTP latency, so in order to keep the software application running and accepting sequential calls it should be invoked from another VI that doesn't contain loops so that Web service calls would be directed to the proxy VI and the proxy VI would be communicating with the running software application. The communication between the proxy VIs and the main application cannot be local even if both run on the same machine because the proxy VIs will be uploaded to memory, hosted by the LabVIEW *Application Web Server*, and deployed independently from the LabVIEW environment. Therefore, the communication will be realized on network using "Shared Variables" as illustrated in Figure 59.

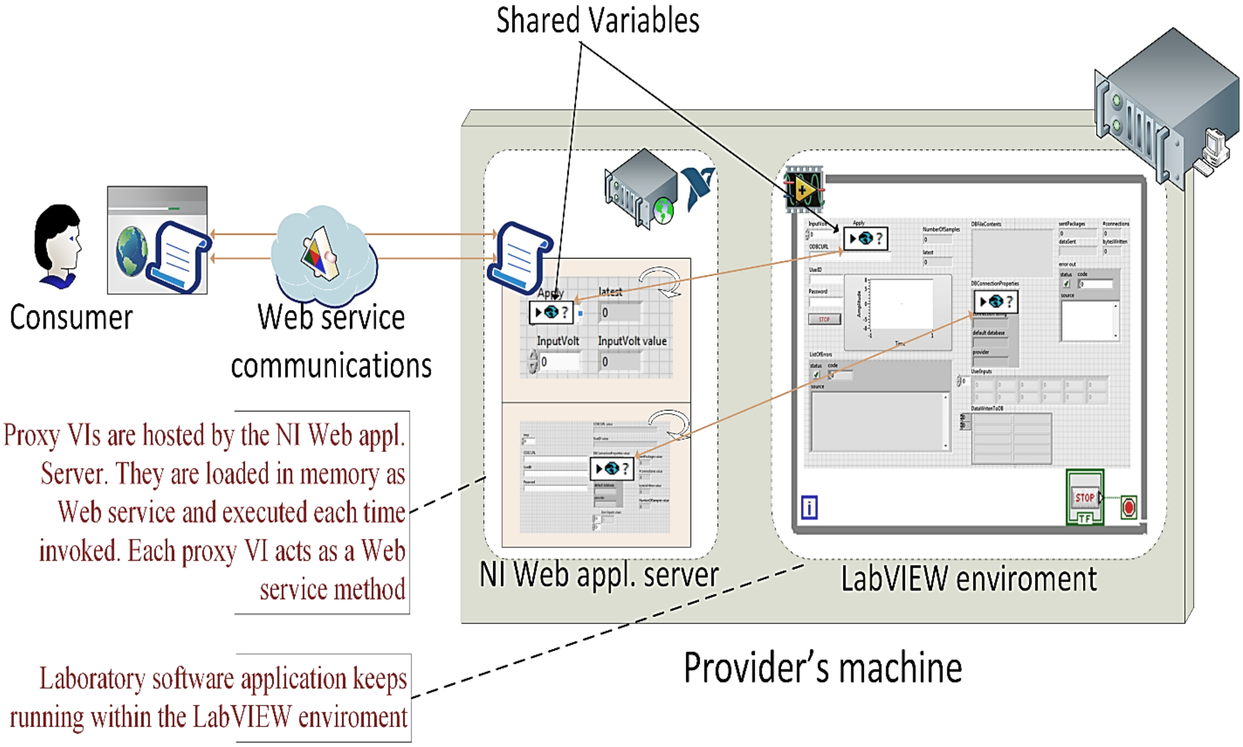

Shared Variables
Consumer
Web service communications
Proxy VIs are hosted by the NI Web appl. Server. They are loaded in memory as Web service and executed each time invoked. Each proxy VI acts as a Web service method
Laboratory software application keeps running within the LabVIEW enviroment
Apply
latest
InputVolt
InputVolt value
NI Web appl. server
LabVIEW enviroment
Provider's machine
STOP

Figure 59. Web service implementation in LabVIEW.

Providers (or developers or administrators) are free to select the input and output variables to be exposed to the proxy VIs as "Shared Variables" and thereby to consumers as Web service. Some variables might be exclusively made available for the administrator. Input variables are configured as "Shared Variables" with "Read Access Mode" and output variables are configured as "Shared Variables" with "Write Access Mode". Each proxy VI will act as a Web service method and will be executed each time invoked. In the current example, two proxy VIs will be needed, *method1* and *method1*. The *method1* VI is used for introducing voltage input and retrieving the latest tacho-meter reading value. The *method2* VI is used for ending the experiment, introducing database credentials and retrieve all the rest of information including list of all input values, database connection information, streaming information, etc. Hereafter, the above mentioned steps are going to be explained in details. The selected variables from the developed software in Figure 58 and Figure 55 are listed in Table 12 (i.e., these variables are case-sensitive).

Table 12. List of selected Global Variables.

	Access Mode	Data Type	Proxy VI	Description
InputVolt	Read	Double	method1	Input voltage value
Apply	Read	Boolean	method1	Apply input value
stop	Read	Boolean	method2	Ends the iteration of the read/write while loop, and the two streaming while loops
latest	Write	Double	method1	Latest tachometer reading
NumberOfSamples	Write	Long	method2	Number of samples written by the DAQ virtual channel
sentPackages	Write	Long	method2	Total number of TCP packages sent during the streaming
#connections	Write	Long	method2	Total number of TCP connections realized
ODBCURL	Read	String	method2	Path of the user's database .dsn file
UserID	Read	String	method2	User's ID
Password	Read	String	method2	User's password
DBConnectionProperties	Write	Cluster	method2	Database connection properties
UserInputs	Write	Array	method2	Total input voltage values by user

All the "Shared Variables" should be network published and a single element real time FIFO is enabled in order to store "Shared variables" data until new is introduced. Otherwise, buffering can be used to store data. Once the main software application starts, the "Shared Variables" are automatically deployed and made published to the network. The administration and monitoring of the "Shared Variables" are realized through the "NI Distributed System Manager". Several modifications have been done in the developed software in Figure 55, mainly the following: replacing input controls with "Shared Variables" and assigning indicators to monitor the introduced values (e.g., the Apply input control was replaced by a led button indicator); the entire code was wrapped into a "While Loop" in order to keep the software running continuously for several sessions; and finally, some modifications were realized in the streaming code in order to enable Web service to read only the total values (e.g., total sentPackages are read at the end of the iteration). The new modified code including the "Shared Variables" is shown in Figure 60.

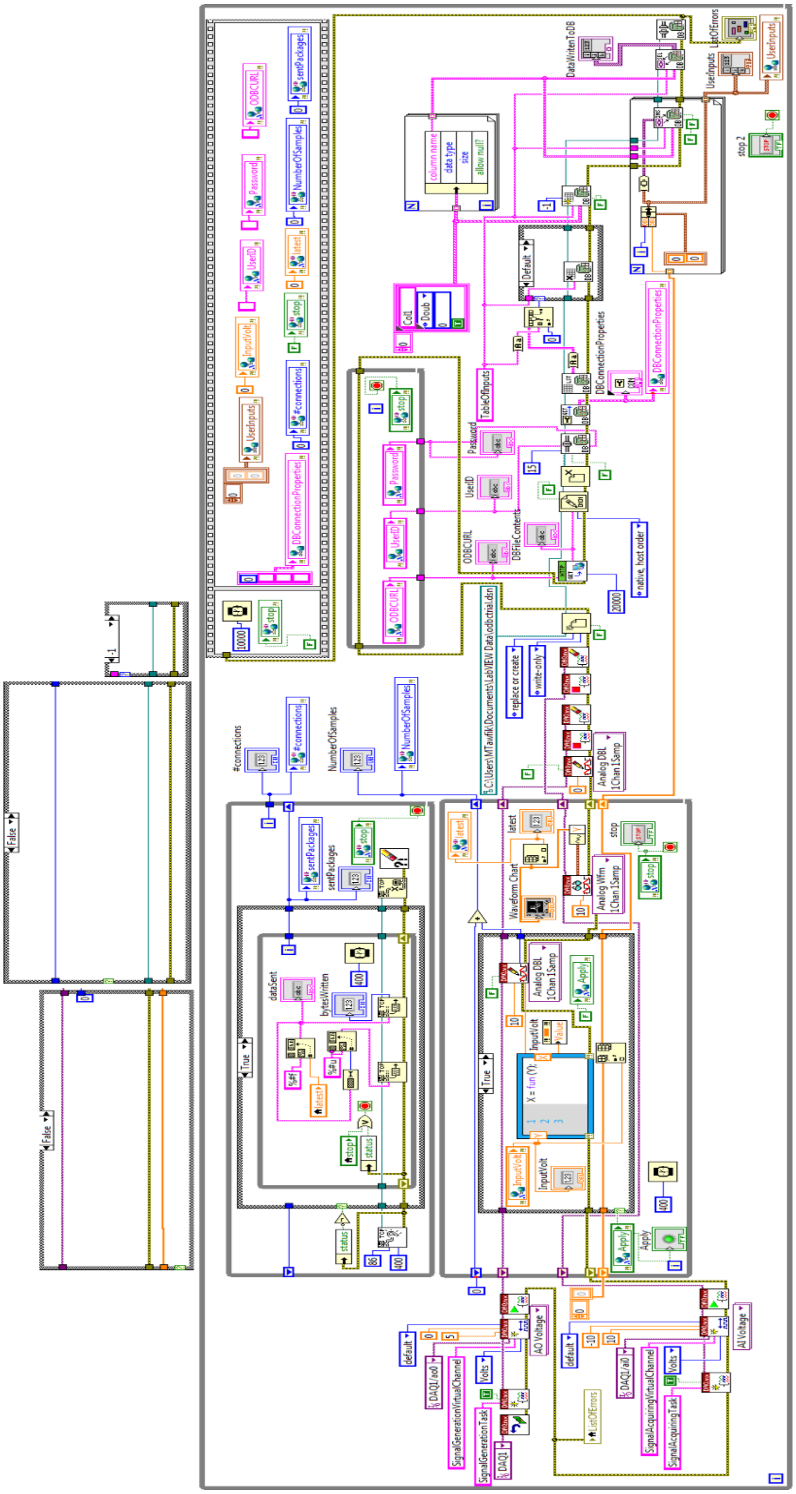


Figure 60. Block diagram of the motor-tacho laboratory's software (with Shared Variables).

Afterwards, the created “Shared Variables” are used to create the two proxy VIs, method1 and method2. The first proxy VI, *method1.vi,* is shown in Figure 61.

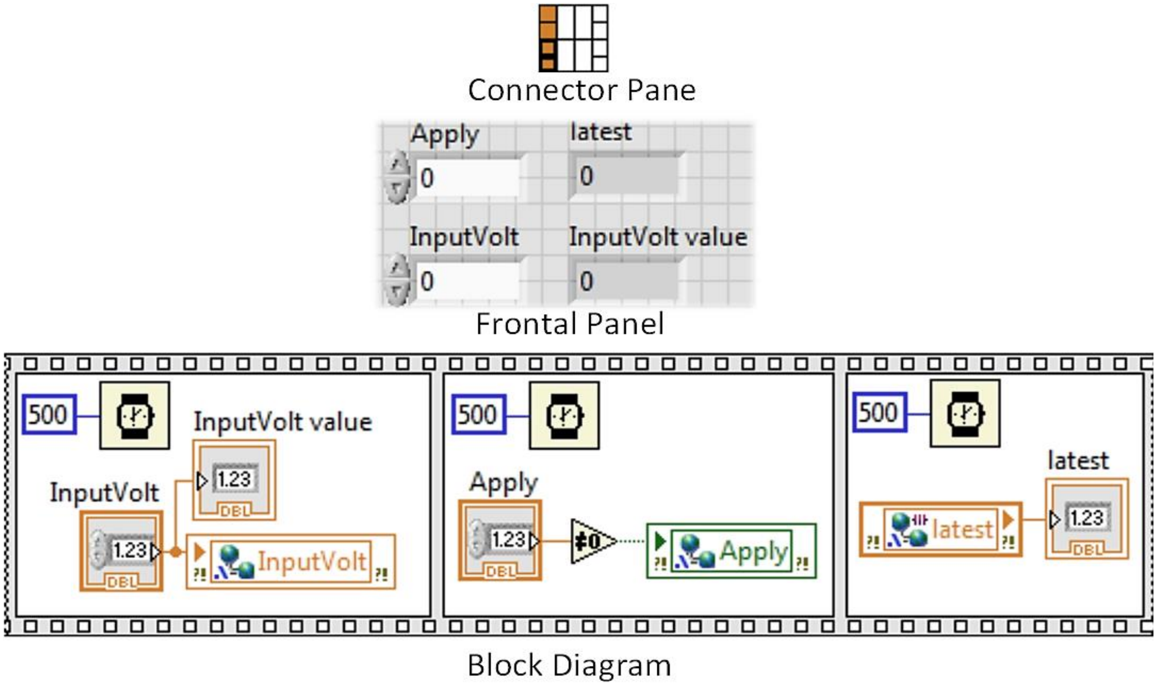
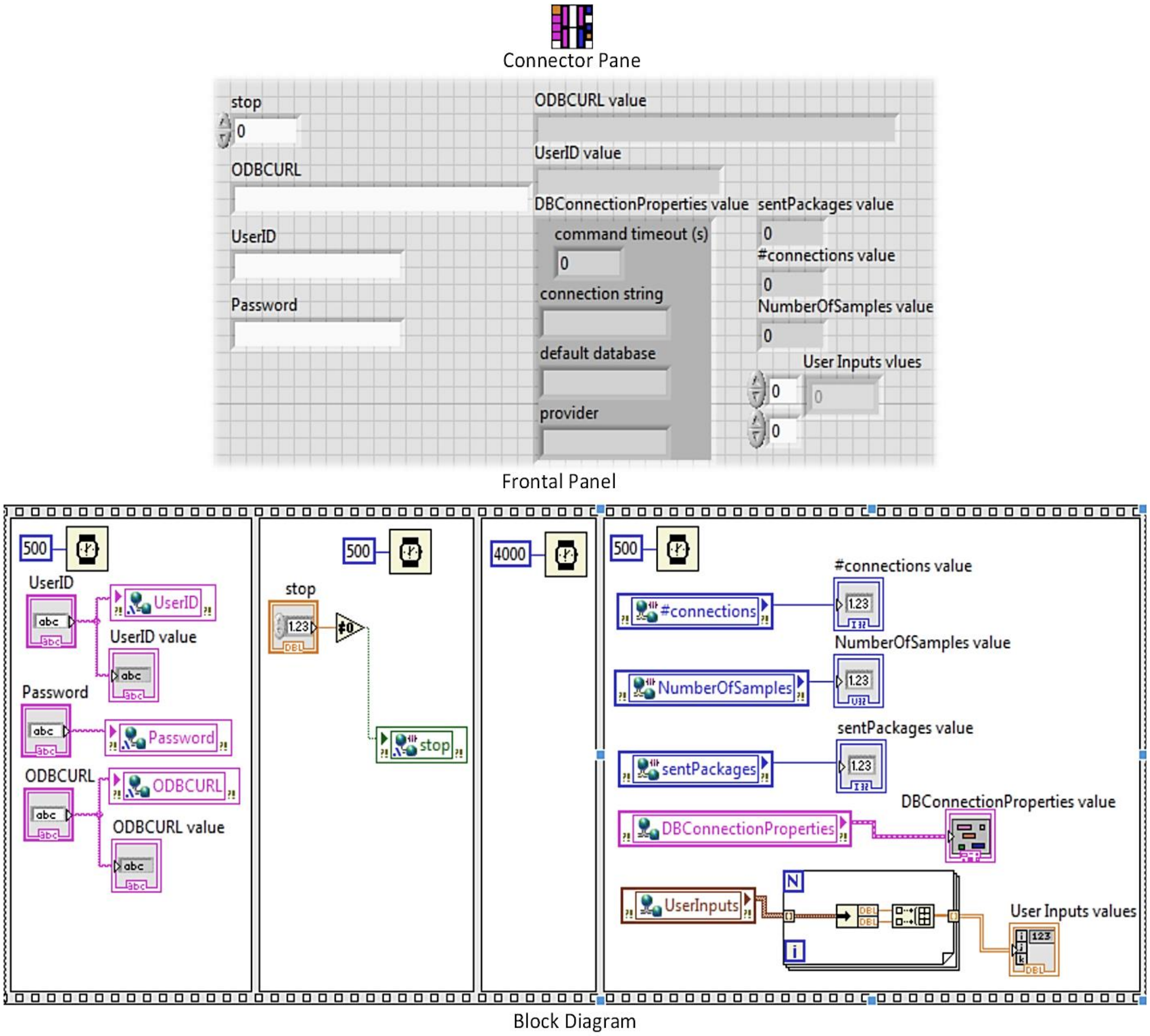


Figure 61. method1.vi.

As seen in Figure 61, using *method1.vi* user can insert the input volt value, apply the change, and read the tachometer latest value. Each “Shared Variable” used are connected to either a control or an indicator depending on its “Access Mode”. Additionally, an indicator is added to the “InputVolt” in order to allow user to visualize the value he/she inserted, but in this case a different name is assigned to that indicator (“InputVolt value” instead of “InputVolt”) to avoid conflicts. The input and output terminals of the “Frontal Panel” are connected to the “Connector Pane” in order to be recognized by the LabVIEW Web Service tool, which is going to be discussed shortly. Owing to the inherent latency of Web service, the sequential execution of the code is configured by the “Flat Sequence Structure” and each sequence is retained for a reasonable amount of time. By trial and error and by testing using several calls, the total method retention is selected to be 1500 ms, which is partitioned and shared by all the sequences. This retention value was sufficient to give response in the correct order at each time. The sequence starts with inserting the value, then applying the change (i.e., all “Shared Variables” are restored to their Default value in the main application VI), and finally, retrieving the result. A final remark is that in LabVIEW, Web service doesn’t work with Boolean controls. Therefore, in order to pass “TRUE” or “FALSE” value to “Apply”, a “Not Equal To 0?” function is used, so that any numeric value except zero will be interpreted as “TRUE”. The second proxy VI, *method2.vi*, is shown in Figure 62.


Connector Pane

Frontal Panel

Block Diagram

Figure 62. method2.vi.

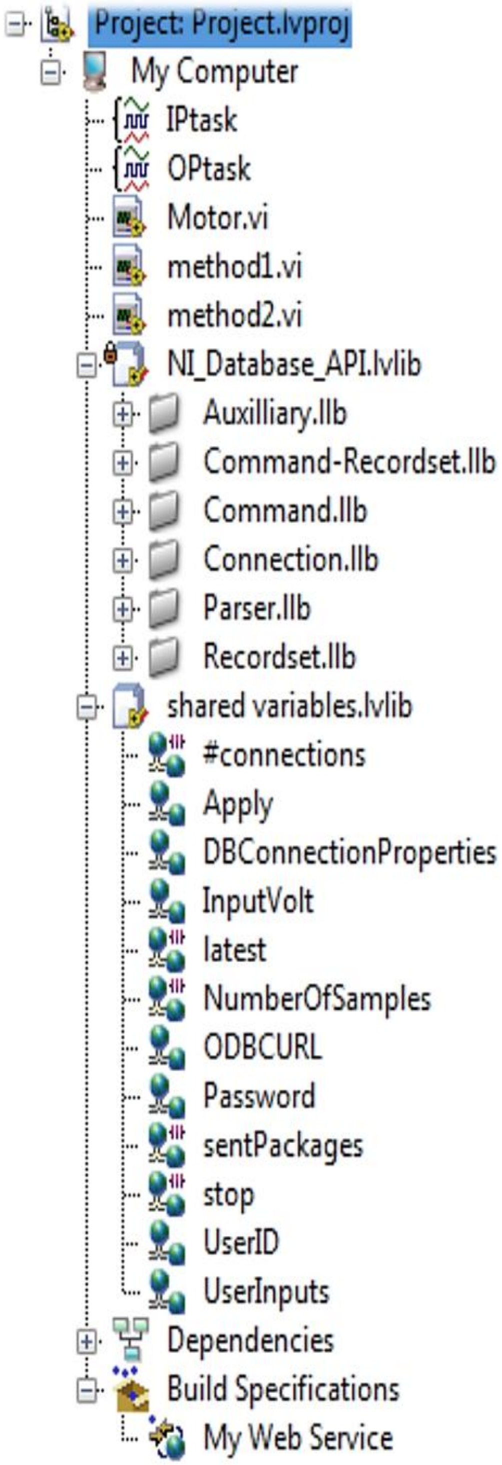
As seen in Figure 62, *method2.vi* allow user to stop the running experiment session and introduce his/her ODBC-compliant database address along with his/her database credentials in order to get a copy of the complete list of the voltage input values in his/her database. In response, the user gets back: the introduced parameters by him/her except the password value, a list of voltage input values, database connection information, number of TCP connections and sent packages during the tachometer reading streaming, and number of samples written by the virtual DAQ channel. The sequence starts with introducing input values, then stopping the session and its loops, and finally reading the output values. Notice that a reasonable retention time of 4000 ms was added between stopping the session and reading the values in order to make sure that all parameters are updated before reading—owing to the inherent latency of the network and the "Shared Variables". Since all the parameters are restored to their Default values at the end of the session, a retention time must be added and it must be greater that the reading retention (4000ms) so that reading can occur before Default value restoration. The selected time was 10000 ms as shown in Figure 60. The output of the "UserInputs" variable is an array of clusters, which is not possible to be viewed using Web service, therefore it was transformed into a pure array of integer values.

The next step is to build the RESTful Web service of the developed software using the "LabVIEW Web Service Tool". This is done from the project items panel, *Project:projectname.lvproj>Build Specifications>New>Web Service(RESTful)*. First, we define the name of the Web service and the destination directory at the hosting machine. Afterwards, we choose the source files of the Web service methods (i.e., in this case, the *method1.vi* and the *method2.vi*, and not the main application VI as this will be running independently in the LabVIEW environment itself as demonstrated in Figure 59). The output format type is chosen to be XML. The most important configuration is the "URL Mappings" as it defines the form of each Web service method calls. The "Override Terminal Default" option allows assigning a default input value each time we initiate a Web service call. All the valid controls that are correctly named and wired in the VI using the "Connector Pane" should appear in the "VI Terminal" of the "Override Terminal Default" panel so that user can assign default values. Indicators don't appear but if they are valid they will appear automatically in the Web service response message. In the URL mapping, the input terminal parameters that correspond to the controls in the “Frontal Panel” are preceded by a Colon “**:**” and they accept any single fragment value (number or text, or both). In order to assign multiple fragments to a single string input, the colon should be replaced by an Asterisk “*****”—in this case, the terminal parameter must only occur at the end of the URL. This technique is useful for passing file paths (e.g., as in case of the ODBC URL) or other hierarchical values using a custom browsing URL. The URL mapping of the two HTTP methods, *method1* and *method2*, are shown in Table 13.

Table 13. URL Mappings of the two HTTP methods, method1 and method2.

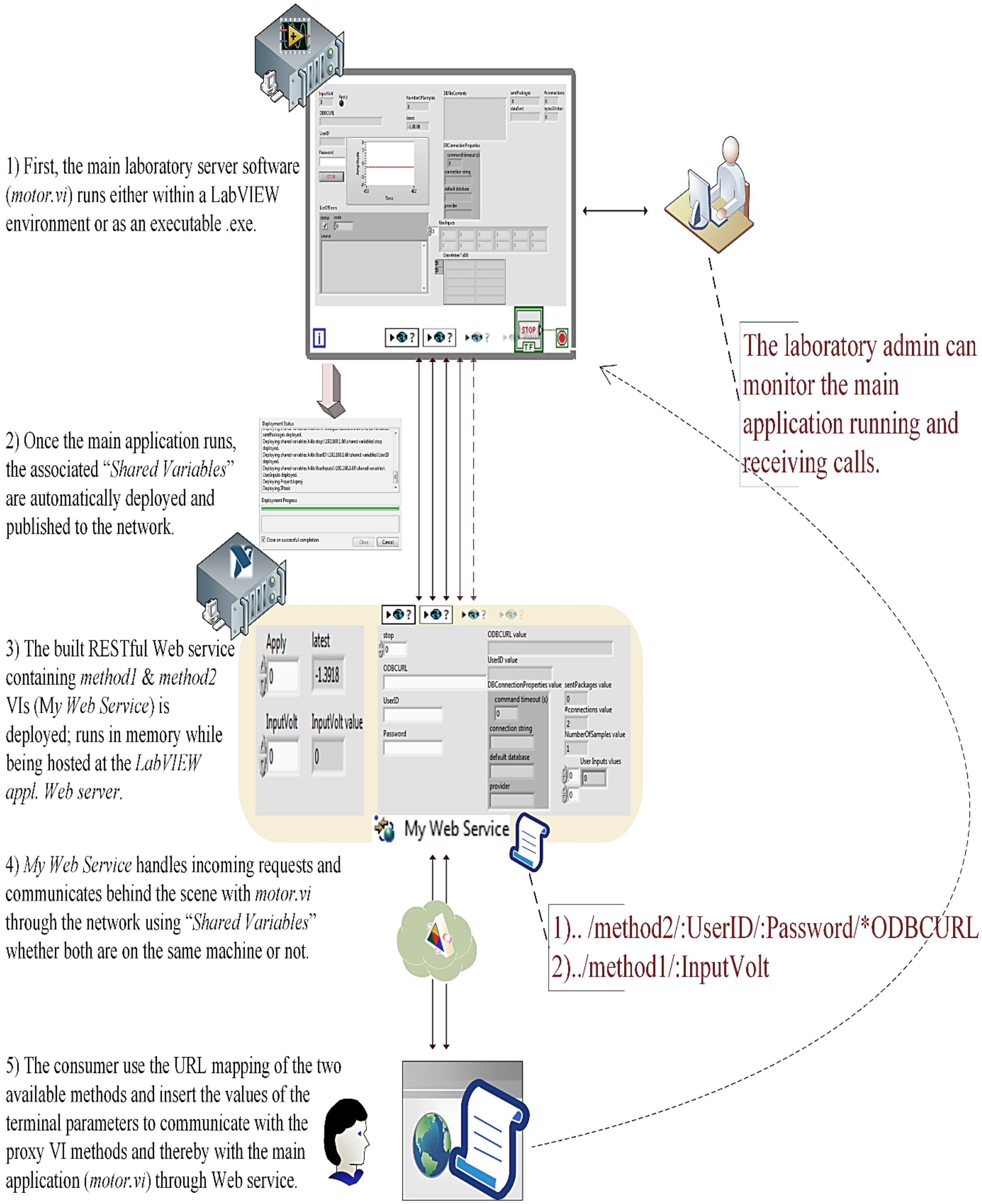
	Web Method VI	**HTTP Method**	**Default Override**
/method2/:UserID/:Password/*ODBCURL	method1.vi	GET	Apply (value=1)
/method1/:InputVolt	method2.vi	GET	stop (value=1)

Once the Web service is built and deployed, it will be hosted at the “NI Application Web Server” and loaded in memory as long as the server service is running. Since Web service internally (in the hosted machine or in the network) communicates with the laboratory server main software application using “Shared Variables”, the laboratory server main software application should be running in a LabVIEW environment (or in an executable .exe application) and its “Shared Variables” should be deployed and published on the network in order to receive updates from the proxy method VIs that are deployed as Web service as described in Figure 59. Figure 63 shows the final item panel of this project (Project.lvproj) including the two main DAQ tasks (IPtask and OPtask), the “Shared Variables, the main VI (motor.vi), the two proxy VIs (method1.vi and method2.vi), the “LabVIEW Database Toolkit” APIs, the Dependencies (the MATLAB *fun.m* file) and the created RESTful Web service instance (My Web Service). A detailed topology of Web service implementation in LabVIEW is provided in Figure 64.



- Project: Project.lvproj
  - My Computer
    - IPtask
    - OPtask
    - Motor.vi
    - method1.vi
    - method2.vi
    - NI_Database_API.lvlib
      - Auxilliary.llb
      - Command-Recordset.llb
      - Command.llb
      - Connection.llb
      - Parser.llb
      - Recordset.llb
    - shared variables.lvlib
      - #connections
      - Apply
      - DBConnectionProperties
      - InputVolt
      - latest
      - NumberOfSamples
      - ODBCURL
      - Password
      - sentPackages
      - stop
      - UserID
      - UserInputs
    - Dependencies
    - Build Specifications
      - My Web Service

Figure 63. Final LabVIEW project items panel.



1) First, the main laboratory server software (*motor.vi*) runs either within a LabVIEW environment or as an executable .exe.

2) Once the main application runs, the associated "*Shared Variables*" are automatically deployed and published to the network.

```
Deployment Status
sentPackages deployed.
Deploying shared variables Mlb:stop\\192.168.1.66\shared variables\stop
deployed.
Deploying shared variables Mlb:UserID\\192.168.1.66\shared variables\UserID
deployed.
Deploying shared variables Mlb:UserInputs\\192.168.1.66\shared variables\
UserInputs deployed.
Deploying Project.lvproj
Deploying JPtask
Deployment Progress
☑ Close on successful completion    Close    Cancel
```

3) The built RESTful Web service containing *method1* & *method2* VIs (My *Web Service*) is deployed; runs in memory while being hosted at the *LabVIEW appl. Web server*.

4) *My Web Service* handles incoming requests and communicates behind the scene with *motor.vi* through the network using "*Shared Variables*" whether both are on the same machine or not.

5) The consumer use the URL mapping of the two available methods and insert the values of the terminal parameters to communicate with the proxy VI methods and thereby with the main application (*motor.vi*) through Web service.

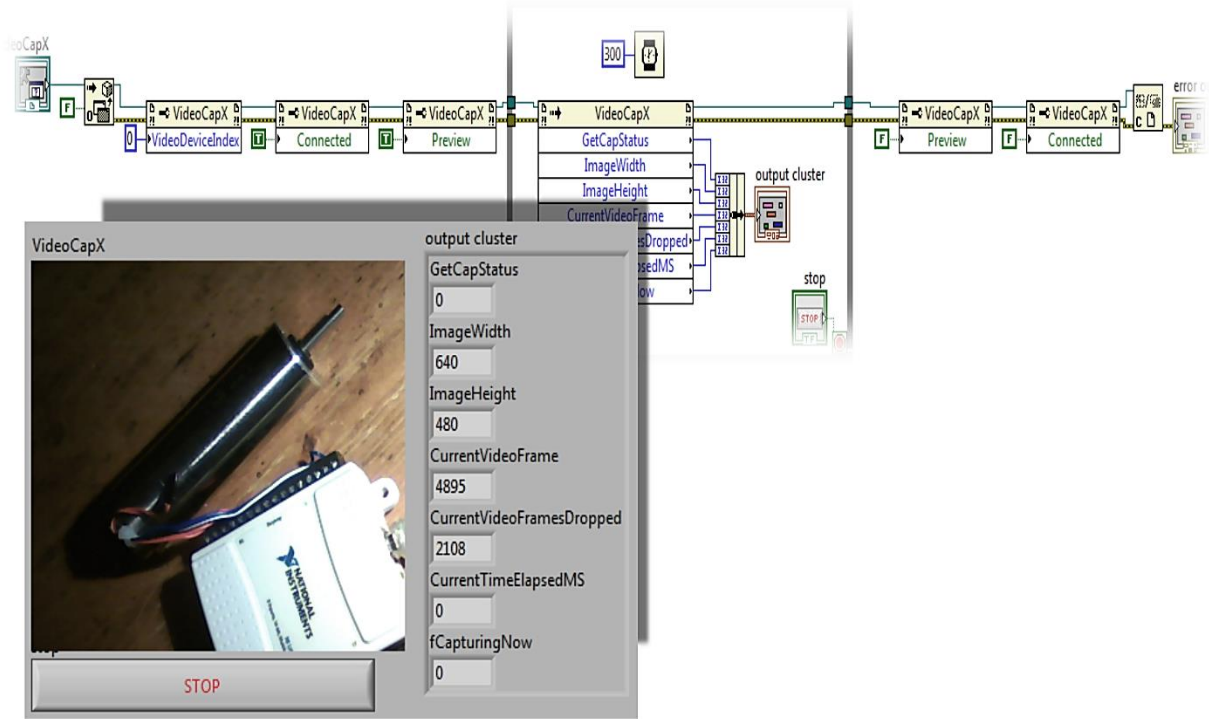
The laboratory admin can monitor the main application running and receiving calls.

1).. /method2/:UserID/:Password/*ODBCURL
2)../method1/:InputVolt

Figure 64.Topology of Web service implementation in LabVIEW.

6.2.3. **Video Streaming Server**

Now that we have the Web service deployed and the TCP streaming for the tachometer reading also configured, the last step is to configure the camera streaming server. There exist several solutions for camera streaming, among them commercial solutions such as VideoCapX (www.videocapx.com). VideoCapX is also distributed as an ActiveX control, which allow users to integrate it in an ActiveX container such as LabVIEW. This is done by initializing a reference to the ActiveX object and creating the sequential methods and properties using the "Property Node" and the "Invoke Node" functions in LabVIEW as shown in Figure 65.



VideoCapX

output cluster

GetCapStatus
0

ImageWidth
640

ImageHeight
480

CurrentVideoFrame
4895

CurrentVideoFramesDropped
2108

CurrentTimeElapsedMS
0

fCapturingNow
0

STOP

Figure 65. Webcam preview using VideoCapX ActiveX control in LabVIEW.

Using additional ActiveX properties and methods, user can get live camera streaming on a network. Another alternative, which is going to be adopted in this example, is to use an open source software such as VLC from VideoLAN (www.videolan.org). VLC allows establishing a camera streaming server over a network with a wide variety of protocols and encoding formats. In this example, we are going to stream an integrated Webcam over HTTP. In order to do this, first we select “Open Capture Device”, then we select the integrated Webcam device, then we choose a reasonable caching value (e.g., 600 ms), then we press the “Stream” button. Now we have the source adjusted in the Media Resource Locator (MRL) as the URI “**dshow://**”. Following this step (the source configuration step), we setup the destination as streaming over HTTP (i.e., the path is left empty so that it refers to the local host server), the streaming port (e.g., TCP 88), and the transcoding profile. User can create his/her own transcoding profile and define the encapsulation format and codecs as long as they are supported by the streaming protocol. The transcoding profile information should be attached with the URI to the laboratory consumer in order to allow him/her to receive correctly the data. Once the streaming server runs, it would be possible to live view the camera capture over HTTP through the URI *http://192.168.1.66:88*, where the 88 is the port and the IP address is the current IP of the machine hosting the camera server. In order check the streaming server, a new VLC instance is opened as a client and from the “Open Network Streaming” panel, and the URI is inserted to view the streaming data live over HTTP as shown in Figure 66.

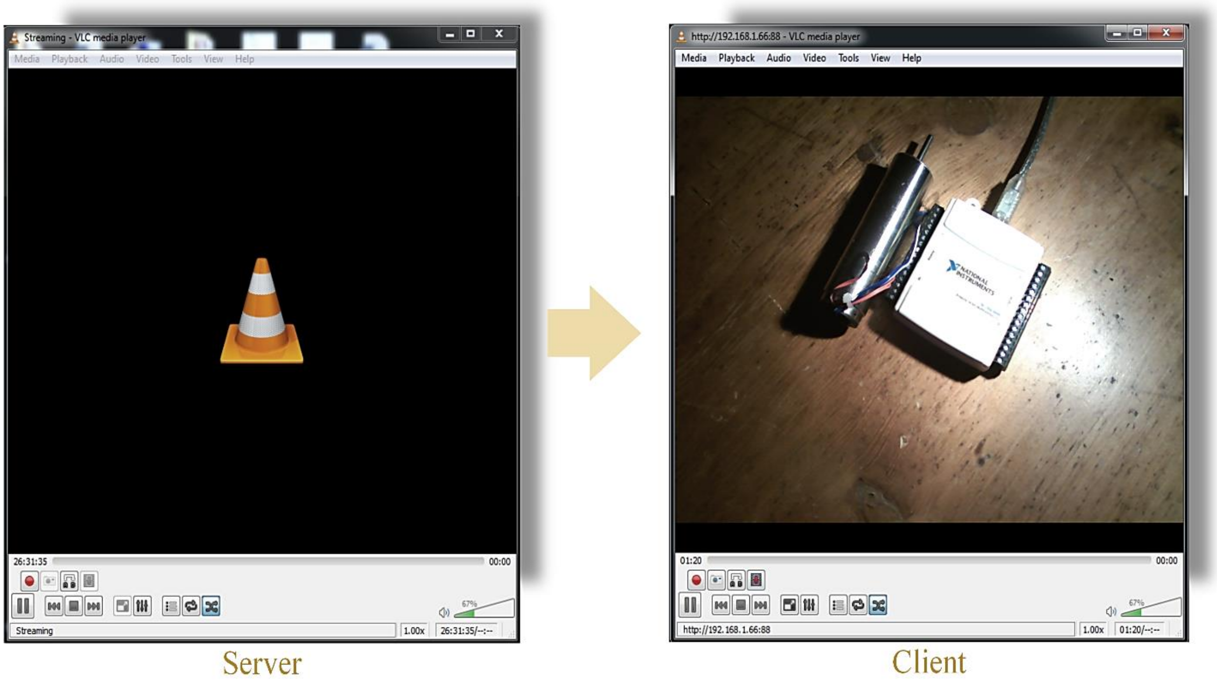

Streaming - VLC media player
Media Playback Audio Video Tools View Help
26:31:35
00:00
67%
Streaming
1.00x
26:31:35/--:--
Server
http://192.168.1.66:88 - VLC media player
Media Playback Audio Video Tools View Help
01:20
00:00
67%
http://192.168.1.66:88
1.00x
01:20/--:--
Client

Figure 66. Video streaming using VLC.

6.3. **Delivery as a Service**

Bringing together the developed components of the modular motor-tacho's remote laboratory, it would be possible to deliver it as a set of abstracted services for consumers to integrate it in their own application independently of the underlying technologies they would prefer to use. In addition to the abstracted services exposed from the modular components, developer can add extra information such as metadata ontologies, experiment description, hours of operation, contact information, etc. The service description file could be delivered as follows:

{LaaS}

Tacho-Motor Remote Laboratory.pdf

Description: A remote laboratory for switching on a permanent magnet DC motor and reading the output using a tachometer.

Keywords: Principles of control, electric machines, DC motor, power engineering, electrical engineering.

Provider: Universidad Nacional de Educación a Distancia (UNED).

Contact: (email) *mtawfik@ieec.uned.es*, (TEL) 0034-000000000.

Operation days/hours: Open for public each week from Friday to Monday from 9am to 9 pm, otherwise contact for enquiry.

Additional resources and manuals: *http://........../Motortacho_User_Manuel.pdf* , *http://youtube.com/watch?V=6..MotortachoRemoteLab*.

Component modules: WebCam (provider), Motor-tacho (provider), and database (user, standard=ODBC).

Provided services:

1) *http://192.168.1.66:88* >>>for Webcam streaming

i.e., Encapsulation ASF/WMV, video codec (VP8), audio codec: MPEG audio, caching 600ms.

2) *http://192.168.1.66:89*>>>for tachometer reading streaming

i.e., data is sent as a String and at 400 ms sampling rate.

3) *http://192.168.1.66:80/webServicemethod1/:InputVolt* >>>for introducing input voltage value.

i.e., as a response, the user gets the latest tachometer reading.

3) *http://192.168.1.66:80/webservice/method2/:UserID/:Password/*ODBCURL*>>> for introducing the path of the user's .dsn file and his/her database credentials.

i.*e., as a response, the session is ended and a list of the input voltage values introduced by the user is copied to his/her database in a new created columns (col1 and col2). Additional information are provided as a response such as: the introduced parameters by him/her except the password value, a list of the voltage input values, database connection information, number of TCP connections and sent packages during the tachometer reading streaming, and number of samples written by the virtual DAQ channel.*

Note that it is up to the consumer how to use these services and it is up to the service broker how to index the LaaS, certainly with regard to the keywords and metadata parameters available.

6.4. Consumption

As mentioned previously, LaaS doesn't contemplate the way consumers would deploy the provided services in their end-user applications. It only assures the delivery of services in an abstracted way and basing on well-known and accepted standards. Figure 66, provided an example on how the Webcam streaming URL could be consumed via VLC. However, consumer is free to implement it in any kind of application container, plugin-based sand box, etc. The consumption of the rest of the services is described hereafter in the following example.

Now let's consider the scenario from the consumer's perspective. After reading the service description file and having understood the laboratory functions and components, the consumer prepares his own ODBC-compliant database as a requirement for the laboratory usage (i.e., not necessarily in this particular and simple example). For instance, if the consumer has the most famous open source database "MySQL", he/she should install its ODBC connector first. The ODBC driver should be listed in the "ODBC Data Source Administrator" (i.e., in Windows OS or its equivalent in other Oss) in order to be allowed to create and configure the associated .dsn file. In this example the .dsn file is called "*odbctrial2.dsn*" and it contains the following information:

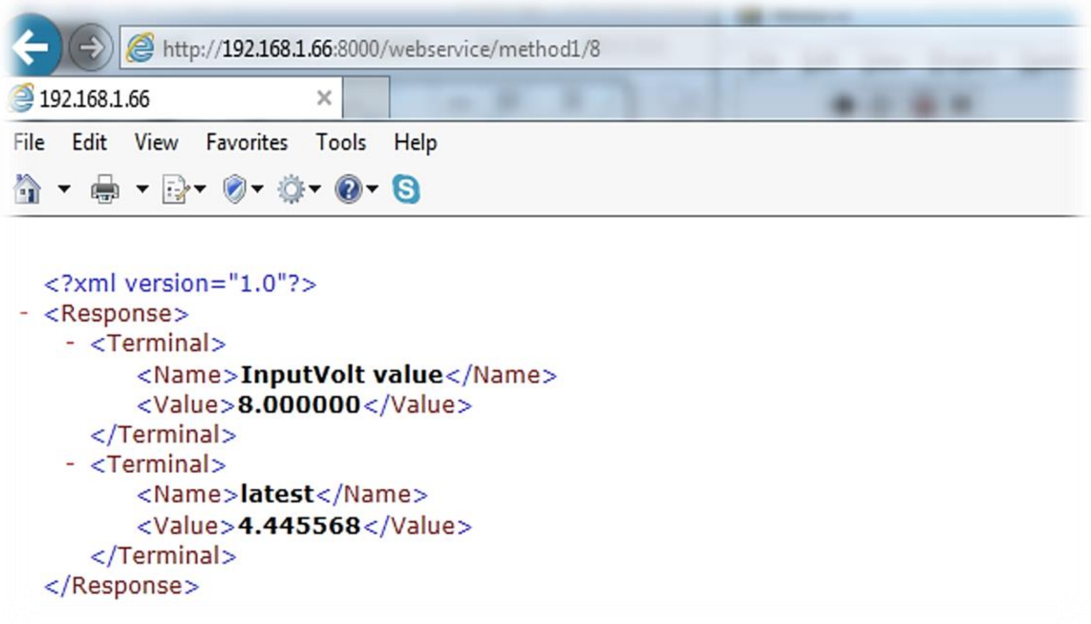
```
[ODBC]

DRIVER=MySQL ODBC 5.2 ANSI Driver

UID=root                  # database user

PORT=3306                 # the port which the database is listening to

DATABASE=labview          # the database table that laboratory would have access to

SERVER=192.168.1.66[1]    # the address of the database hosting server
```

[1] Notice that in this laboratory prototype, the consumer's and the provider's machine IP address are the same because both scenarios are realized on the same physical machine, just for demonstration purpose.

Now that the consumer's database .dsn file is ready, it should be transferred to the provider (or the laboratory server software) to access it and thereby to communicate with the consumer’s database. In this example the transferring method is defined by the provider, which is a URL of the .dsn file. Thus, consumer should host his/her .dsn file in a Web server in order to transfer it as a URL, the path of the file will be *http://192.168.1.66:80/odbctrial2.dsn*.

Finally, we start using the laboratory and realizing the experiment using the provided services, assuming that the provider is already running the “motor.vi” application, deploying the “Shared Variables”, and running the RESTful “My Web Service” instance on the “NI Web Application Server” as previously shown in Figure 64. In other words, assuming that we access within the time period determined by the provider in the service description file. Let’s start the experiment with the following sequences:

1) We introduce a voltage input value of 8V using the method1 as shown in Figure 67. As a result the value is applied (i.e., but only 5V due to the implemented control strategy), the tachometer latest reading is retrieved, and the motor will keep rotating.



http://192.168.1.66:8000/webservice/method1/8

```
  <?xml version="1.0"?>
- <Response>
  - <Terminal>
        <Name>InputVolt value</Name>
        <Value>8.000000</Value>
    </Terminal>
  - <Terminal>
        <Name>latest</Name>
        <Value>4.445568</Value>
    </Terminal>
  </Response>
```

Figure 67. RESTful Web service response to method1.

2) During the execution of the motor, we will stream the tachometer reading using the provided URL (http://192.168.1.66:89). For example, we can stream it using LabVIEW as shown in Figure 68.

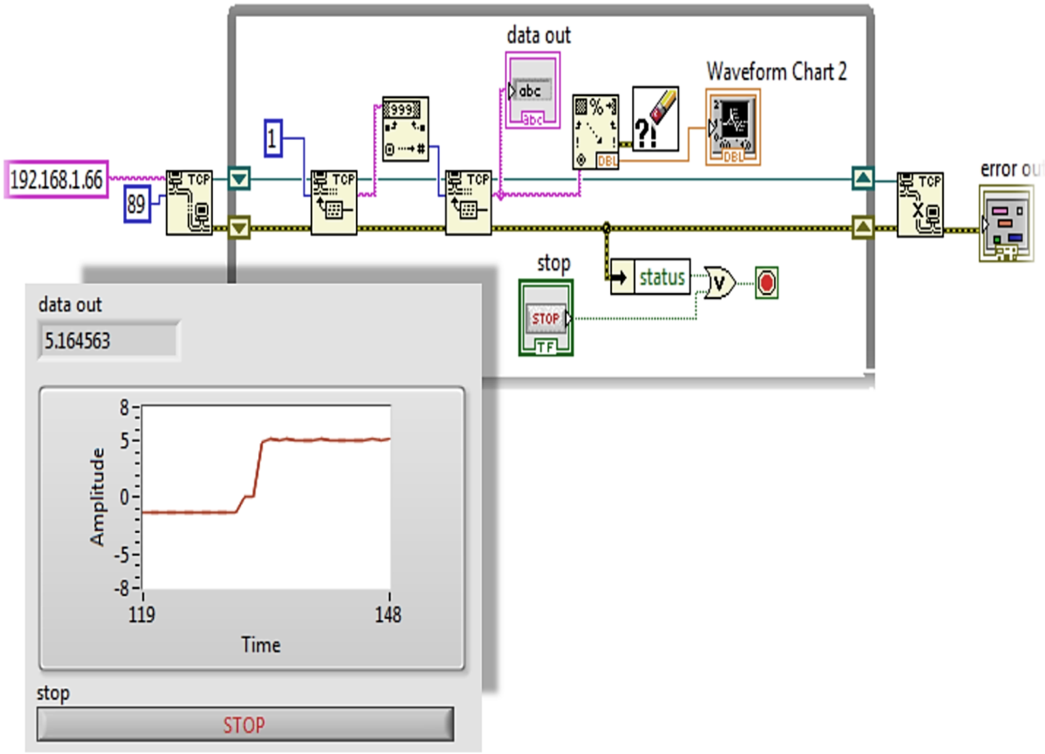


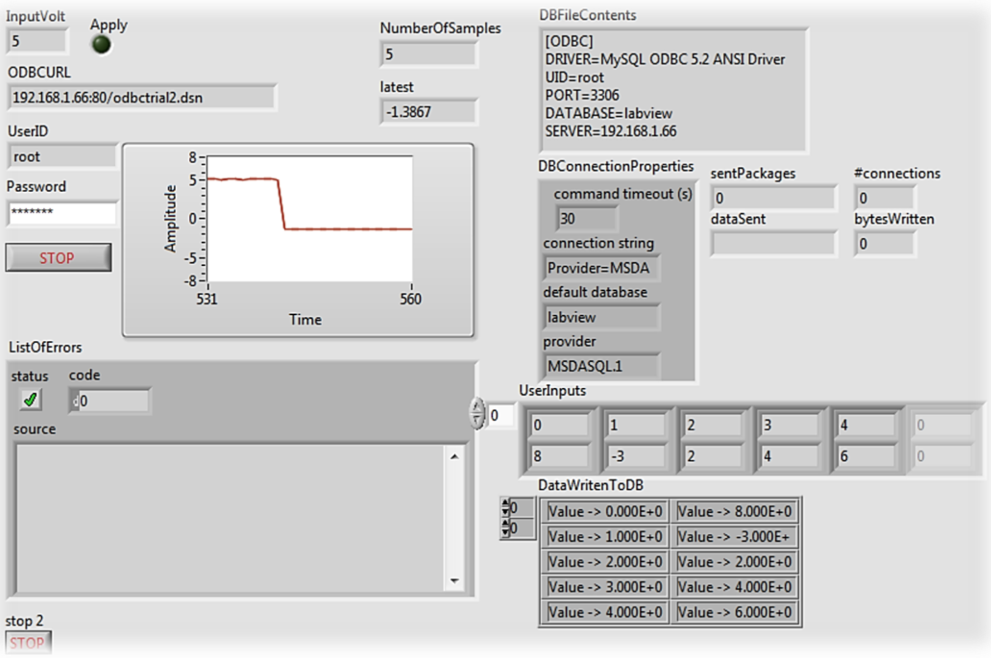
Figure 68.Streaming of tachometer reading.

In the code shown in Figure 55 a TCP connection is opened using the provided URL, then the input data package size is retrieved first and converted to a number using the “Decimal String to Number” function , then the data package is read. The data is retrieved in the same format it was sent (String format) then it’s converted to a Double using the “Scan from String” function and visualized by a “Waveform Chart”.

3) Step 1 is repeated with different input values: -3, 2, 4, and 6V. Notice that the -3V value will apply only 0V.

4) Finally, using Web service method2, we stop the application session and the motor execution, we send the database .dsn file to the laboratory server software through the URL (http://192.168.1.66:80/odbctrial2.dsn), and we insert the database credentials (username = root and password = labview).

i.e., note that during the execution of the experiment sessions, the administrator or the lab provider can monitor the software running (the main application *motor.vi*) and that the main application should be always running otherwise the calls of the proxy VIs (implemented as Web service) wouldn't get response and consequently wouldn't answer the user's petitions. Figure 69 shows the response of step 4 at the provider's main application, the *motor.vi*.



InputVolt: 5

ODBCURL: 192.168.1.66:80/odbctrial2.dsn

UserID: root

Password: *******

NumberOfSamples: 5

latest: -1.3867

DBFileContents:

```
[ODBC]
DRIVER=MySQL ODBC 5.2 ANSI Driver
UID=root
PORT=3306
DATABASE=labview
SERVER=192.168.1.66
```

DBConnectionProperties

command timeout (s): 30

connection string: Provider=MSDA

default database: labview

provider: MSDASQL.1

sentPackages: 0

dataSent:

#connections: 0

bytesWritten: 0

ListOfErrors

status: ✓ code: 0

source:

UserInputs: 0

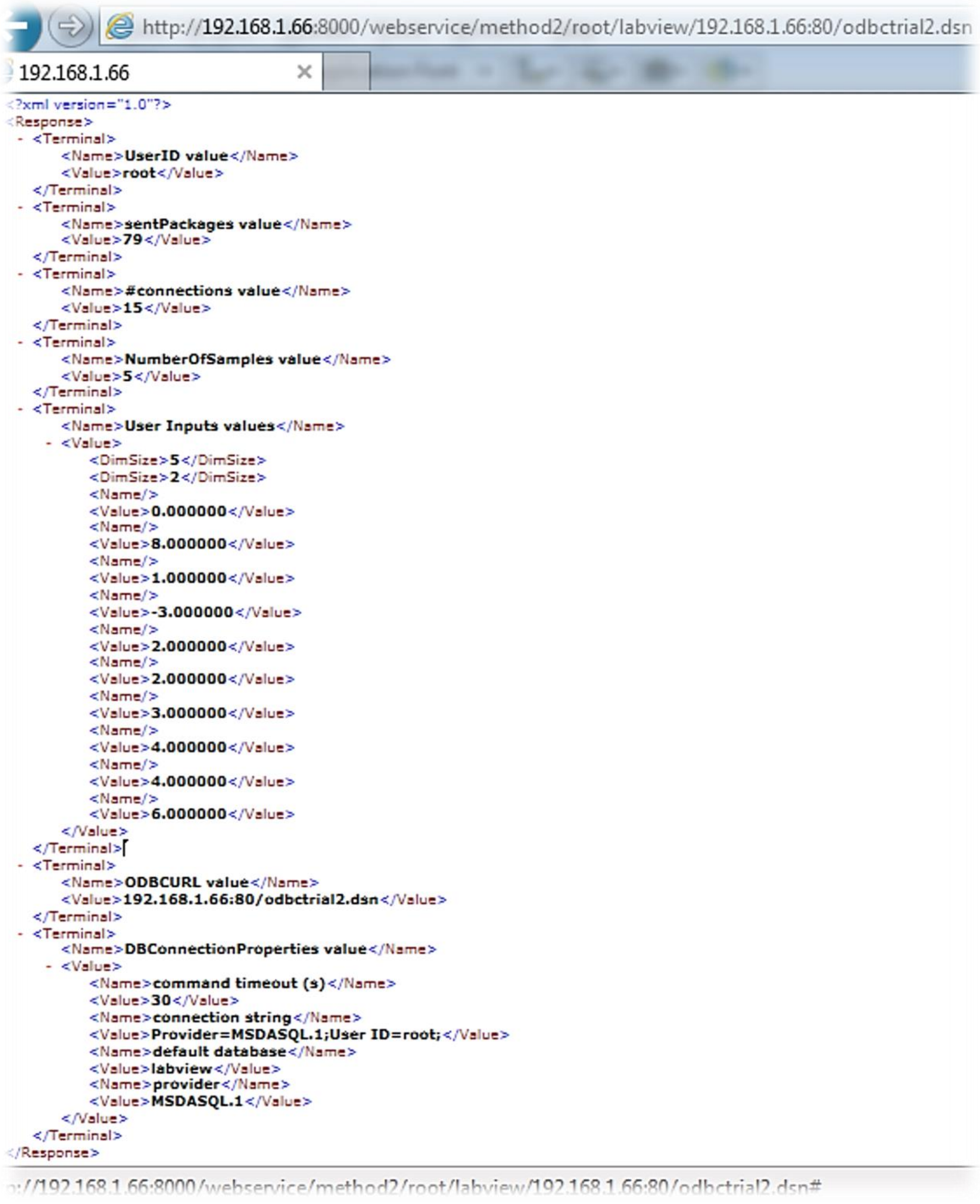
0	1	2	3	4	0
8	-3	2	4	6	0

DataWritenToDB: 0, 0

Value -> 0.000E+0	Value -> 8.000E+0
Value -> 1.000E+0	Value -> -3.000E+
Value -> 2.000E+0	Value -> 2.000E+0
Value -> 3.000E+0	Value -> 4.000E+0
Value -> 4.000E+0	Value -> 6.000E+0

Figure 69.Response to method2 at the motor.vi .

The response of step 4 (to the consumer) is shown in Figure 70.



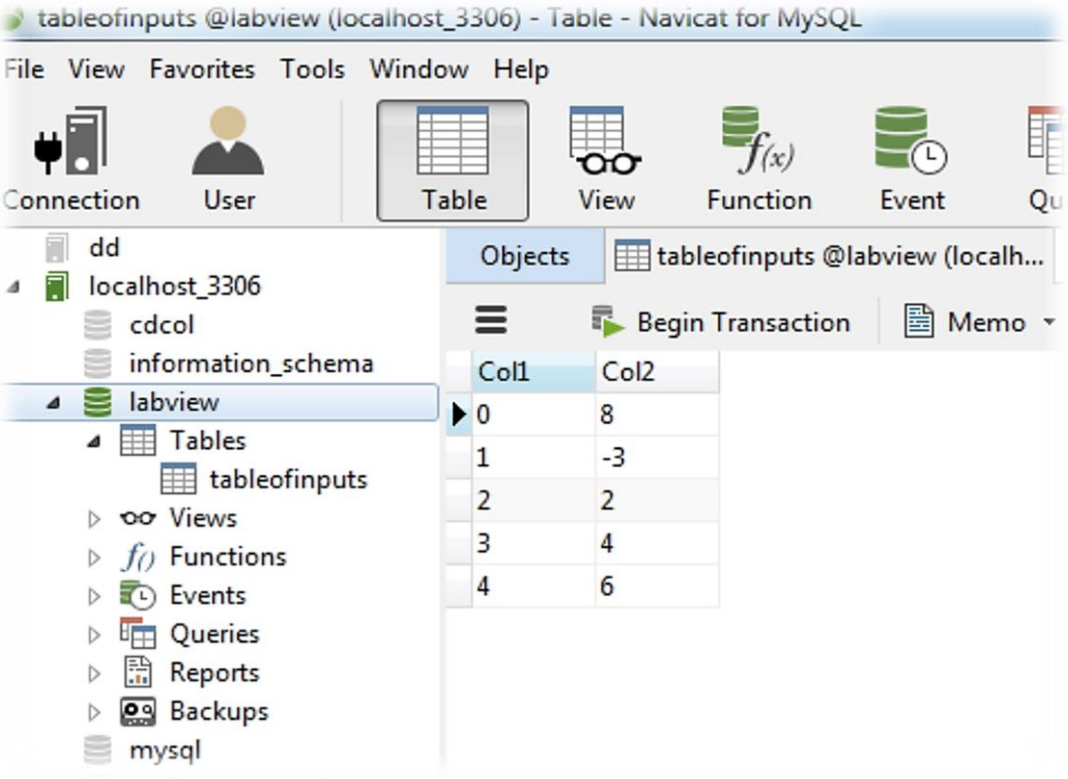
http://192.168.1.66:8000/webservice/method2/root/labview/192.168.1.66:80/odbctrial2.dsn

```
<?xml version="1.0"?>
<Response>
 - <Terminal>
      <Name>UserID value</Name>
      <Value>root</Value>
   </Terminal>
 - <Terminal>
      <Name>sentPackages value</Name>
      <Value>79</Value>
   </Terminal>
 - <Terminal>
      <Name>#connections value</Name>
      <Value>15</Value>
   </Terminal>
 - <Terminal>
      <Name>NumberOfSamples value</Name>
      <Value>5</Value>
   </Terminal>
 - <Terminal>
      <Name>User Inputs values</Name>
    - <Value>
         <DimSize>5</DimSize>
         <DimSize>2</DimSize>
         <Name/>
         <Value>0.000000</Value>
         <Name/>
         <Value>8.000000</Value>
         <Name/>
         <Value>1.000000</Value>
         <Name/>
         <Value>-3.000000</Value>
         <Name/>
         <Value>2.000000</Value>
         <Name/>
         <Value>2.000000</Value>
         <Name/>
         <Value>3.000000</Value>
         <Name/>
         <Value>4.000000</Value>
         <Name/>
         <Value>4.000000</Value>
         <Name/>
         <Value>6.000000</Value>
      </Value>
   </Terminal>
 - <Terminal>
      <Name>ODBCURL value</Name>
      <Value>192.168.1.66:80/odbctrial2.dsn</Value>
   </Terminal>
 - <Terminal>
      <Name>DBConnectionProperties value</Name>
    - <Value>
         <Name>command timeout (s)</Name>
         <Value>30</Value>
         <Name>connection string</Name>
         <Value>Provider=MSDASQL.1;User ID=root;</Value>
         <Name>default database</Name>
         <Value>labview</Value>
         <Name>provider</Name>
         <Value>MSDASQL.1</Value>
      </Value>
   </Terminal>
</Response>
```

p://192.168.1.66:8000/webservice/method2/root/labview/192.168.1.66:80/odbctrial2.dsn#

Figure 70. RESTful Web service response to method2.

5) Finally we check the create columns (col1 and col2) and the data written to our database (the array of input voltage values; the 5 inserted values) as shown in Figure 71.



Col1	Col2
0	8
1	-3
2	2
3	4
4	6

Figure 71. Data written to the database.

6.5. **Summary**

In this chapter, a simple practical LaaS deployment example was provided as a proof-of-concept. A modular motor-tacho laboratory was developed and delivered as a loosely coupled services, according to the LaaS paradigm, to be consumed within any application container independently of the underlying technology adopted in both. The idea of its experiment is very simple as it aims to emphasize the LaaS concept rather than delving into the technical details of the experiment. Despite its simplicity, it includes many features that are common in complex applications—in order to proof the concept reliably—such as hybrid programming using LabVIEW and MATLAB, database access, and video streaming. LabVIEW was used for developing the laboratory server software and

MATLAB was used for creating a control strategy. The laboratory has three component modules and one of these modules, the database, adheres to a standard connector, ODBC. This laboratory requires that the consumer connects his/her own ODBC-compliant database to retrieve the list of input voltage values. Finally, a service description file was created, which includes: a description of the laboratory and the experiment; an ODBC connector for the database, RESTful Web services calls supported by the laboratory; video streaming URI; and tachometer measurement streaming over TCP URI.

Chapter 7

7. Conclusions

This thesis introduced a novel paradigm, LaaS, for developing modular remote laboratories—based on independent component modules—and implementing them as a set of loosely-coupled services to be consumed with a high level of abstraction and virtualization.

LaaS implies the development of remote laboratories as component modules and these components all together are delivered in form of a set of loosely-coupled services to be consumed by users. LaaS strongly relies on well-known industrial and Web standards and middleware technologies for its entire communications and inter-communications. LaaS follows SOA in terms of defining the relation between laboratory providers, laboratory consumers, and service broker or repository in which remote laboratories are registered and indexed under metadata standards and ontologies. LaaS merges features from cloud computing—in terms of consuming services on-demand with minimum restrictions and higher virtualization—and features from grid computing—in terms of global distribution. LaaS embraces the WoT in terms of coupling with heterogeneous services and bringing objects to the Web for all spectrum of needs—in either formal or informal contexts.

The purpose and the overall goals of the proposed LaaS paradigm can be summarized in the following points: (1) define an organized manner for sharing remote laboratories globally among institutions; (2) allow wrapping remote laboratories in any heterogeneous application container (e.g., widget, applet, or any Web client) independently of the underlying technology adopted in both, as well as, their coupling and mashing up with heterogeneous services (e.g., learning objects) across the Web; (3) facilitate maintenance, reusability, and leveraging legacy equipment; (4) allow interchangeability of components between provider and consumer—seamlessly and programmatically— insofar as consumer could contribute with one or more component instead of the fully-reliance on the provider’s equipment and facilities; (5) promote online experimentation and discovery in either every day’s formal or informal contexts; and (6) set principles for a first global standardized design pattern—for remote laboratories development and implementation—to be followed and adopted by remote laboratories developers.

For clarity, two demonstrative theoretical examples were provided in Section 5.6. In the first example, the owner provides his/her modular laboratory as a set of services according to the LaaS and the consumer allocates the laboratory in the marketplace and downloads the service description file. Using the file, the consumer builds his own widget application and wraps it in a SCORM container in order to be able to load it in any SCORM-compatible LMS and further couple it with any local widget. The second example is similar but the provider there doesn’t wish to share some of his/her laboratory component modules, the database and the power supply, in order to reduce the load on his/her own equipment. The provider instead leaves it to the consumer to connect his/her own modules as long as they adhere to the same adopted standard (i.e., ODBC connector for the database and IVI/VISA connector for the power supply).

Finally, the proposed theory was applied to the real world and a simple practical LaaS deployment example was provided. A modular motor-tacho laboratory was developed and delivered as a loosely-coupled services, according to the LaaS paradigm, to be wrapped in any application container independently of the underlying technology adopted in both. The idea of its experiment was very simple as it aimed to emphasize the LaaS concept rather than delving into the technical details of the experiment. Despite its simplicity, it included many features that are common in complex applications—in order to proof the concept reliably—such as hybrid programming using LabVIEW and MATLAB, database access, and video streaming. LabVIEW was used for developing the laboratory server software and MATLAB was used for creating a control strategy. The laboratory has three component modules and one of these modules, the database, adheres to a standard connector, ODBC. This laboratory requires that the consumer connects his/her own ODBC-compliant database to retrieve the list of input voltage values. A service description file was created, which includes: a description of the laboratory and the experiment; an ODBC connector for the database, RESTful Web services calls supported by the laboratory; video streaming URI; and tachometer measurement streaming over TCP URI. Finally, the provided services were consumed and results were obtained.

7.1. Suggestions for Future Works

In this thesis, the LaaS paradigm and the modular remote laboratories concept have been clearly described and proven to be affordable and feasible with the current available technologies and industrial and Web standards.

From the low level perspective, future works should be focused on expanding the application range and modularizing different kinds of remote laboratories with different operation scenarios in order to investigate further issues and discover further improvements. Modularization entails primarily replacing conventional connections with standardized counterparts and adding input and output connectors to each component in an abstracted manner in order to allow maximum flexibility and interchangeability.

From the high level perspective, efforts should be focused on the structure of the market place and the way LaaSs will be made available to the public. This primarily involves an appropriate scheduling or queuing mechanism. The assumption of “either the laboratory is currently available or not” was made, being the simplest solution and it can be implemented in many ways. For example, using a Web service call to check if the laboratory is available or a session is currently running by another user. This was demonstrated in the prototype described in Chapter 6, where no new session is accepted unless no previous session is currently running. In addition, the provider still has the ability to grant access to certain IPs basing on previous agreements. However, these techniques wouldn’t suffice for managing a large scale deployment with numerous groups and students. Even though the major focus in this thesis was on the low level side, while the high level side was left to the consumers, someone might argue that the provider is still inherently involved in any transaction and therefore an appropriate scheduling mechanism should be supported at the low level side and initially considered in the future design. Thus, the upcoming challenge would be implementing a scheduling mechanism using extra layers while maintaining the service description file as abstracted as possible in accordance with the premise of the LaaS paradigm.

7.1.1. Standardization

The final goal of the LaaS paradigm is to set bases towards an acceptable model to which developers and laboratory providers could adhere to. For this goal, the LaaS paradigm was devised basing on a wide study and analysis of the cons and pros of all the previous efforts and approaches aiming to integrate and implement remote laboratories efficiently. In other words, it meant to assure the optimum scenario. At the same time the context of the next generation Web and learning environments was taken into consideration. An initial proposal was provided for a loosely coupled and modular-based remote laboratories that can be built and implement with a high level of visualization and abstraction. This proposal was described in Chapter 5 and clarified by two theoretical examples. Afterwards, it was proven affordable by the practical example presented in Chapter 6.

However, it is not possible to define a standard without a wide agreement among involved partners. Having accepted this concept, only few efforts would be required to achieve an initial standard proposal draft. The main tasks should be centered in the following:

- Defining a standardized format for the service description file including the presentation for of each section.
- Agreement on a standard metadata and even on determined ontologies to be adopted.
- Agreement on the widest spread Web and industrial standards (e.g., ODBC, IVI/VISA, RESTFul, Websockets, etc.) to be adopted.

This could be achieved with the support of the IEEE P1876™ Standard for Networked Smart Learning Objects for Online Laboratories Working Group, the GOLC consortium, and interested partners from industry such as National Instruments and Microsoft.

7.2. Summary of Outputs

The major outputs of this thesis can be summarized and classified into three main categories as follows:

7.2.1. Research

Two novel concepts were developed and introduced: (1) modular remote laboratories, which aims to convert laboratories into modular components in order to facilitate maintenance, reusability, and interchangeability of components seamlessly and programmatically; and (2) LaaS paradigm: which aims to convert modular remote laboratories into a set of services to be consumed by users with a high level of abstraction and virtualization. It defines, as well, the broader implementation mechanism of these laboratories. LaaS paradigm allows overcoming common concurrent challenges in remote laboratories developing and implementation such as inter-institutional sharing, interoperability with other heterogeneous systems, coupling with other services and learning objects, difficulty of developing, and standardization. In order to proof the affordability and the feasibility of the proposed solution, a prototype was developed and tested and results were obtained. (Chapter 5 & 6)

The aforementioned conducted research entailed the realization of the following studies:

1. A study on remote laboratories applications in all disciplines of electrical engineering education. The study targets the development and implementation problems and the limitations associated with different kinds of applications (e.g., low power applications, real-time applications, etc.).
2. A study on the scenarios of integrating remote laboratories with different types of educational systems, addressing the strengths and weaknesses of each solution.

3. A study on the characteristics of the more likely to be the Web of tomorrow; the next generation learning environment, and the possible methods of accessing online learning resources in the future.
4. A study on the available middleware technologies for SOA implementation.
5. A study on the generic architecture of remote laboratories for all kind of disciplines and on all its components.
6. A study on the standard communication buses and connectors.
7. A study on the available technologies for developing laboratory server software.
8. A study on the Web client and user interface technologies.

7.2.2. Development and Implementation

The following have been developed and implemented:

- A set of remote electronic circuits' experiments were developed using VISIR. It is targeted to undergraduate engineering curricula and it has been deployed in official undergraduate engineering degrees at UNED. Deployment results has been obtained. As well, it has been deployed in the world first remote lab-based MOOC on electronic circuits, which is delivered by UNED to the public. (Chapter 3)
- A set of new advanced remote electronic circuits' experiments has been developed using VISIR. It is oriented to postgraduates and apprentices and addresses labor markets and industrial needs in order to diminish the gap between academia and workplace. Industrial-related issues are emphasized in the design to allow understanding the behavior of electronics components. This kind of experiments hasn't been previously reported in the literature either in conventional lab or remote lab arrangement. This combination, as a result, converted VISIR system into a unique training platform of its kind for remote electronics experiments. This set of experiments has been deployed in an inter-institutional European online master degree program in ICS. This first-of-its-kind master degree program is conducted by 5 European institutions from 4 different

European countries and each partner contributes with at least a subject and a remote laboratory access. (Chapter 3)

- A Modular remote laboratory prototype was developed and results were successfully obtained. (Chapter 6)

7.2.3. Dissemination and Publications[1]

- Authored 4 journal articles (i.e., journals includes *IEEE Industrial Electronics Magazine* and *IEEE Transactions on Learning Technologies*).
- Co-authored 4 journal articles (i.e., journals includes *IEEE Journal of Latin-American Learning Technologies* and *Solar Energy*).
- Authored a research book chapter and co-authored another research book chapter.
- Authored 16 conference papers and co-authored another 19 conference papers (i.e., conferences includes *Society for Information Technology & Teacher Education International Conference-SITE*, *ASEE Annual Conference & Exposition*, *IEEE Global Engineering Education Conference-EDUCON*, and *IEEE/ASEE Frontiers in Education Conference-FIE*).
- Further journal articles and conference papers have been submitted and currently under review.

7.2.4. Other Merits[2]

- Realized a 6 month research visit at University of Technology, Sydney (UTS), Sydney, Australia.
- Realized a 5 month research visit at École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland.
- Participated in 10 research projects (i.e., 3 national, 5 European, and 2 international).

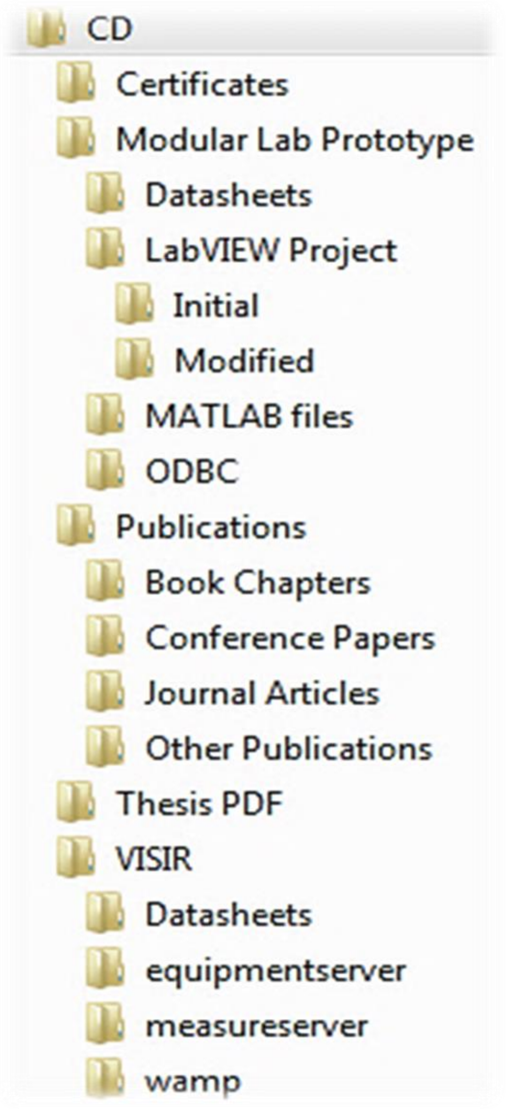
[1] The majority of the publications were presented at/published in international and prestigious conferences and journals.

[2] A detailed list of accomplishments and contributions is found in Section 1.7.

- Obtained the “best student paper award” at EDUCON 2012, Marrakesh, Morocco.
- Served as a reviewer in 3 international and prestigious journals (International Journal of Engineering Education-IJEE, IEEE Network, and Computer Applications in Engineering Education) and in 4 international and prestigious conferences (FIE 2012, FIE 2013, EDUCON 2012, and TALE 2012).

Appendix A

A. CD Content



- CD
  - Certificates
  - Modular Lab Prototype
    - Datasheets
    - LabVIEW Project
      - Initial
      - Modified
    - MATLAB files
    - ODBC
  - Publications
    - Book Chapters
    - Conference Papers
    - Journal Articles
    - Other Publications
  - Thesis PDF
  - VISIR
    - Datasheets
    - equipmentserver
    - measureserver
    - wamp

This page intentionally left blank.

Appendix B

B. Resumen (Español)

B.1. Introducción

Los laboratorios siempre han sido un elemento clave e indispensable en la educación de ingeniería eléctrica. No hay duda que la experimentación, sobre todo en ingeniería y ciencias aplicadas, tiene un papel muy importante en la observación de los fenómenos y en consolidar el entendimiento de los conceptos teóricos. Según los criterios generales de la acreditación de los programas de ingeniería 2013-2014 de la organización no gubernamental ABET (el acrónimo ingles de "Accreditation Board for Engineering and Technology"), las competencias prácticas son esenciales para asegurar el cumplimiento de los objetivos educativos de los programas. Esto quedó presente en la lista del tercer criterio: (1) la capacidad de diseñar y realizar experimentos, así como de analizar y de interpretar los datos; y (2) la capacidad de utilizar las técnicas, las habilidades, y las herramientas modernas y necesarias para las prácticas de ingeniería.

Sin embargo, en la literatura, poca atención se ha prestado a las actividades prácticas en comparación con los otros asuntos pedagógicos. Desde los primeros años, y tras la aparición de la educación formal de ingeniera, el enfoque se basó en cómo lograr un equilibrio entre la teoría y la práctica. Con el tiempo, los criterios de evaluación académicos predominantes han empezado a reconocer primariamente la productividad de la investigación en vez de reconocer las contribuciones a la educación universitaria, lo cual causó menos interés en las actividades que requieren espacio, tiempo, y preparación como el desarrollo de laboratorios para la enseñanza. Los laboratorios son generalmente costosos (i.e., en términos de tiempo y presupuesto) y difíciles de desarrollar, adquirir, administrar, y mantener. Los laboratorios tienden a tener bajas tasas de utilización y su utilización suele ser limitada a unos cursos específicos sin la posibilidad de compartirlos entre las universidades a pesar de su alto coste. El acceso restringido y la inflexibilidad de operación actual de los laboratorios, además de la disminución de los presupuestos y el incremento del número de estudiantes, han hecho presión y han obligado a las universidades a buscar una alternativa eficiente a los laboratorios tradicionales. Esto ha sido posible con la llegada de los laboratorios en línea y la comodidad que proporcionan.

Los laboratorios en línea son aquellos laboratorios a los que se puede acceder y manipular en línea. Hay dos tipos, laboratorios virtuales y laboratorios remotos. Los laboratorios virtuales están basados en simulación y pueden ser o bien una aplicación software independiente o bien una aplicación Web, y no se tratan de equipos físicos o reales. La simulación consistirá en una operación en función del tiempo de un modelo matemático que emula el comportamiento de un sistema físico. La simulación se utiliza en muchos contextos como prácticas, educación, y entretenimiento. Durante la Segunda Guerra Mundial, el simulador de vuelo “Link Trainer” se utilizaba para mejorar la seguridad y reducir el tiempo de formación de más de 500.000 pilotos, ahorrando millones de dólares y muchas vidas. En educación, la simulación se utiliza para modelar sistemas con fin de entender su operación y comportamiento. Es una herramienta muy útil sobre todo para ilustrar un fenómeno en áreas difíciles de visualizar e imaginarse como por ejemplo en: nanotecnología, electromagnetismo, química y física, y ciencias aplicadas. La simulación, sin embargo, ha sido criticada por ser rígida y surrealista, y porque no representa adecuadamente los sistemas reales y su comportamiento. Generalmente, se acepta que la simulación hoy en día no puede sustituir completamente a los laboratorios físicos tradicionales pero podría ser muy efectiva si se utiliza en conjunto con ellos.

Por otro lado, los laboratorios remotos están basados en equipos físicos controlados, monitorizados, y manipulados por Internet. Los laboratorios remotos proporcionan muchas ventajas significativas que no proporcionan los laboratorios tradicionales como optimización de su utilización, compartimiento entre las instituciones con fin de reducir costes, más seguridad, y acceso no limitado ni por consideraciones geográficas ni temporales. Los laboratorios remotos aparecieron en los años 90. A finales de los años noventa, el lanzamiento del servidor de Internet versión (6i) de LabVIEW fomentó y facilitó el desarrollo y la expansión de los laboratorios remotos entre las universidades y en todo el mundo.

Los defensores de los laboratorios en línea abogan en favor de las ventajas asociadas en términos de coste, seguridad, y optimización de uso, mientras que los defensores de los laboratorios tradicionales abogan en favor del manejo físico de los equipos que va en línea con la teoría del conocimiento constructivista—los defensores de los laboratorios remotos podrían discutir que hoy en día los procesos industriales suelen ser automatizados y completamente controlados de forma remota. Muchos debates y estudios comparativos y empíricos se han realizado para determinar cuál de los dos formatos es más eficiente que el otro. La conclusión final descubrió que los resultados del aprendizaje dependen de las instrucciones dadas al grupo de estudiantes y de la colaboración entre sí y con su tutor, independientemente del formato del laboratorio. Por tanto, los desarrolladores de los laboratorios remotos empezaron a contemplar otros factores pedagógicos como su integración en los sistemas educativos y su acoplamiento con otros objetos de aprendizaje.

B.2. Motivación

En la última década, hemos presenciado una proliferación significativa de los laboratorios remotos en todas las áreas de educación de ingeniería eléctrica gracias a la revolución exponencial de las tecnologías digitales. En la primera era del desarrollo de los laboratorios remotos, los esfuerzos se centraban en expandir su rango de aplicaciones y en otros asuntos comunes como seguridad, sistema de reserva, y ancho de banda. En gran medida, muchos de estos problemas han sido resueltos.

Los problemas actuales engloban cuestiones relacionadas con el impacto pedagógico y con la manera de desplegar los laboratorios remotos en la educación. Estas cuestiones incluyen con la integración en sistemas y servicios heterogéneos (como los objetos de aprendizaje) con fin de crear un entorno educativo enriquecido—en lugar de ser monolítico con un diseño fijo—y, por otro lado, con fin de promover la compartición de los recursos entre las instituciones—y por lo tanto poder conseguir una mayor disponibilidad y una reducción del coste.

Las iniciativas realizadas en respuesta a estas necesidades han conseguido proporcionar soluciones para sistemas particulares en vez de ser genéricas y como consecuencia cada institución aprobó su propia solución, que a menudo no funcionaría con otros sistemas de distintas instituciones. Como resultado de esas limitaciones, todavía no es típico que dos universidades compartan sus laboratorios a no ser que lo hayan tenido en cuenta previamente en la etapa del desarrollo y del diseño de sus laboratorios para que permitan este funcionamiento. Del mismo modo, todavía no es típico que una institución tenga a todos sus laboratorios integrados en un sistema único; cada uno de los laboratorios desarrollados hoy en día suele seguir un modo de integración excepcional y distinto por falta del factor de interoperabilidad en el diseño de su arquitectura, por falta de un modelo estandarizado para los desarrolladores a seguir, y por la ambigüedad de los resultados del aprendizaje.

B.3. Solución Propuesta

En respuesta a estas cuestiones, se ha realizado un estudio a fondo sobre las soluciones actuales y las ventajas y desventajas de cada una, así como, sobre las características del supuesto futuro Web, los frutos entornos educativos en línea en los que se deberán desplegar los laboratorios remotos, y los métodos posibles de acceso. Esto con fin de determinar la técnica ideal para desarrollar e implementar los laboratorios remotos de próxima generación de manera muy eficaz teniendo en cuenta consideraciones tanto técnica como pedagógica.

Como resultado, se ha desarrollado un nuevo paradigma, llamado Laboratorio como Servicio (en inglés, “Laboratory as a Service”, y cuyas siglas en inglés son, LaaS), para desarrollar e implementar laboratorios remotos modulares y que se ha propuesto en esta tesis. LaaS pretende abordar los desafíos comunes y actuales en el desarrollo e implementación de los laboratorios remotos como la compartición inter-institucional, la interoperabilidad con otros sistemas heterogéneos, el acoplamiento con otros servicios y objetos educativos, la dificultad de desarrollo, y estandarización.

LaaS implica el desarrollo de los laboratorios como componentes modulares y estos componentes en conjunto se proporcionan en forma de un conjunto de servicios de débil acoplamiento para ser consumidos por los usuarios con un nivel alto de abstracción y virtualización. LaaS depende de estándares industriales y de Web muy conocidos y aceptados por la mayoría para todas sus comunicaciones y para las inter-comunicaciones entre sus componentes. LaaS sigue el modelo de Arquitectura Orientada a Servicios y define la relación entre los proveedores, los consumidores, y el mediador. LaaS combina características de: Computación en la Nube—en cuanto a la consumición de servicios a petición con las mínimas restricciones y la máxima virtualización; Computación Grid—en cuanto a la distribución global; y Web de las Cosas—en cuanto al acoplamiento con los servicios existentes y al traer los objetos a la Web para todo el espectro de necesidades en contextos tanto formales como informales.

B.4. Objetivos

Los objetivos finales son:

1) Definir una manera organizada de compartir los laboratorios remotos globalmente entre las instituciones con las mínimas restricciones y la máxima abstracción y virtualización.
2) Permitir la integración de los laboratorios remotos en cualquier aplicación heterogénea, así como, su acoplamiento con cualquier servicio heterogéneo existente en la Web
3) Facilitar el mantenimiento y la reutilización de los equipos antiguos.
4) Permitir el intercambio de los componentes entre el proveedor y el consumidor de tal manera que el consumidor podría contribuir con un componente o más en vez de depender completamente de los componentes y los equipos del proveedor.
5) Promover la experimentación en línea y el descubrimiento en contextos formales e informales y sin restricciones geográficas ni temporales.
6) Establecer las bases para un primer modelo global de diseño estandarizado para el desarrollo y la implementación de los laboratorios remotos.

B.5. Metodología

Esta investigación se ha llevado a cabo a través de varias etapas cronológicas y unos factores constantes. Los factores constantes fueron:

1) Asistir a conferencias, talleres, seminarios, y Webinars de ámbito tanto nacional como internacional.
2) Consultar la literatura y las recientes ideas y logros en este campo.
3) Participar en proyectos de investigación relacionados con el tema, tanto como en consorcios como grupos de interés.
4) Realizar visitas a otras universidades y a otros grupos de investigación.
5) Publicar y presentar los resultados y los logros científicos en revistas y conferencias prestigiosas y de alta difusión, tanto de ámbito nacional como internacional, así como recibir opiniones y feedback.

Estos factores me mantenían constantemente al tanto con los nuevos desafíos y tendencias durante todo el periodo de investigación. Así como, me ayudaron a definir y afinar los objetivos de mi investigación. Junto con estos factores constantes, la realización de esta tesis pasó por seis etapas cronológicas principalmente.

La primera etapa consistió en una primera experiencia con un laboratorio remoto a través de la instalación del laboratorio remoto VISIR (en inglés, Virtual Instrument Systems in Reality) en la UNED (i.e., siendo la segunda universidad en España en adquirirlo y la sexta a nivel mundial), en el Departamento de Ingeniería Eléctrica, Electrónica, y de Control (DIEEC), y su aplicación en algunas asignaturas oficiales del departamento, y más tarde en el primer Curso en Línea Masivo y Abierto (sus siglas en inglés, MOOC), a nivel mundial, en electrónica basada en laboratorios remotos.

Tras la experiencia adquirida en la implementación, la segunda etapa consistió en expandir esta experiencia e investigar otro tipo de aplicaciones en otras disciplinas de ingeniería y otros temas técnicos como la integración.

Tras el estudio teórico, la tercera etapa consistió en experimentar otros sistemas educativos emergentes de laboratorios remotos como iLab y Sahara y definir sus actuales problemas y los próximos desafíos.

En la cuarta etapa se desarrolló la premisa de esta tesis y se maduró durante la estancia de investigación en Universidad de Tecnología de Sídney (UTS).

La quinta etapa consistió en expandir el rango de aplicación de VISIR y en desarrollar unos experimentos remotos y únicos de electrónica industrial para su implementación en un máster Europeo inter-institucional a distancia basado en experimentación con laboratorios remotos—esto fue uno de los logros principales de esta tesis.

La sexta y última etapa consistió en escribir esta tesis y desarrollar un prototipo durante la estancia de investigación en la Escuela Politécnica Federal de Lausana (EPFL).

B.6. Estructura de la Tesis

Esta tesis se compone de 7 capítulos principales como se muestra a continuación:

- **Capítulo 1, Introducción:** En este capítulo se presenta una introducción genérica sobre el aprendizaje de la ingeniería y las tecnologías educativas. Después, se presenta una introducción sobre el tema planteado en esta tesis incluyendo los problemas abordados, la solución propuesta, los objetivos finales, y la metodología de investigación. Al final, se detallan los resultados y los logros del trabajo elaborado durante los años de desarrollo de esta tesis.

- **Capítulo 2, Aplicaciones de Laboratorios Remotos en el Aprendizaje de Ingeniería Eléctrica:** Este capítulo presenta un estudio sobre el estado del arte de los laboratorios remotos y sus aplicaciones en el aprendizaje de la ingeniería eléctrica. En este estudio se subrayan las aplicaciones más destacadas en cada una de las disciplinas de ingeniería eléctrica. El objetivo de este estudio es recopilar información sobre los problemas y las limitaciones asociados con el desarrollo y la implementación de laboratorios remotos para diferentes tipos de aplicaciones, como las aplicaciones de baja tensión, las aplicaciones de tiempo real, etc. Por otro lado, este estudio dio lugar a entender la arquitectura técnica de los laboratorios remotos y a determinar sus componentes comunes independientemente del tipo de la aplicación. Como conclusión de este estudio, se han establecido las bases de la arquitectura de laboratorios remotos modulares definida en el Capítulo 5.

- **Capítulo 3, Desarrollo e Implementación de Experimentos Remotos de Electrónica:** Este capítulo informa sobre el desarrollo y la implementación de dos conjuntos de experimentos remotos de electrónica con VISIR en la UNED. El primer conjunto está dirigido a los programas de grados de ingeniería y fue utilizado en programas oficiales de grado de ingeniería en la UNED. Además, fue utilizado para la creación del primer MOOC, a nivel mundial, en electrónica basado en laboratorios remotos. Este MOOC fue creado por el departamento DIEEC, está actualmente en marcha y está abierto para el público. El otro es un conjunto de nuevos experimentos remotos de electrónica avanzada. Está orientado a los programas de posgrado y de aprendizaje, y a las necesidades de la industria y del mercado laboral con fin de disminuir el hueco que hay entre la formación académica y el mundo laboral. En el diseño de este conjunto de experimentos se enfatizan temas relacionados con la industria que ayudan a entender el comportamiento de los componentes electrónicos en aplicaciones del mundo real. Este tipo de experimentos no se han desarrollado previamente ni en laboratorios tradicionales ni en laboratorios remotos. Esta combinación de experimentos, como resultado, convirtió VISIR en una plataforma única a nivel mundial de experimentos remotos de electrónica. Posteriormente, el sistema fue examinado y se obtuvieron resultados de medidas a distancia desde los experimentos previamente preparados. Por último, el capítulo informa sobre la aplicación del segundo conjunto de experimentos en un máster Europeo inter-institucional a distancia en sistemas de comunicación e información y basado en experimentación con laboratorios remotos.

- **Capítulo 4, Integración en Sistemas Educativos:** En este capítulo se realiza un análisis cualitativo sobre los esfuerzos y las iniciativas actuales para la integración de los laboratorios remotos en diferentes tipos de sistemas educativos, así como para la comunicación entre diferentes sistemas educativos con fin de compartir los laboratorios remotos integrados en ellos. Se abordan las ventajas y desventajas de cada solución y se elabora una comparación genérica, y a continuación una discusión sobre la misma. Este estudio dio lugar a determinar la técnica ideal para desarrollar e implementar los laboratorios remotos de próxima generación eficientemente teniendo en cuenta las consideraciones técnicas y pedagógicas. Como conclusión de este estudio, se han establecido las bases del propuesto paradigma, LaaS, descrito en el Capítulo 5.

- **Capítulo 5, Laboratorio como Servicio (LaaS):** En este capítulo se describe el propuesto paradigma LaaS. Primero, se presenta un resumen sobre los problemas actuales del desarrollo y la implementación de los laboratorios remotos, y sobre los resultados deseados aplicando el propuesto paradigma LaaS. Después, se describe el contexto del paradigma LaaS teniendo en cuenta las características del futuro Web y los entornos educativos de próxima generación, y los posibles métodos de acceso a los recursos educativos en línea en el futuro, con fin de deducir el mejor escenario de implementación de los laboratorios remotos. Después, se describe el propuesto paradigma, LaaS, así como el concepto de los laboratorios remotos modulares—

basados en componentes interoperables—incluyendo su arquitectura y sus componentes comunes, y como se puede implementarla según el paradigma LaaS. Por último, se presentan dos ejemplos demostrativos junto con un resumen del capítulo y del propuesto paradigma.

- **Capítulo 6, Prototipo de un Laboratorio Remoto Modular:** En este capítulo se presenta un ejemplo práctico demostrativo de la aplicación de LaaS. Se ha desarrollado el primer prototipo de un laboratorio remoto modular que se proporciona como un conjunto de servicios de acoplamiento débil según el paradigma LaaS, para su consumo e integración en cualquiera aplicación independientemente de la tecnología adoptada en ambos. El laboratorio desarrollado permite a los consumidores conectar su propia base de datos (siendo un componente modular intercambiable) utilizando un conector estandarizado.

- **Capítulo 7, Conclusiones:** En este capítulo se presenta una conclusión amplia del tema desarrollado en esta tesis, así como sugerencias para trabajos futuros y un resumen de los resultados y logros durante los años del desarrollo de esta tesis.

B.7. Conclusión y Sugerencias para Trabajos Futuros

En esta tesis se ha propuesto y explicado claramente el concepto del paradigma LaaS y los laboratorios remotos modulares y se ha demostrado su viabilidad y su posible aplicación utilizando las tecnologías, y los estándares tanto industriales como los de la Web, disponibles. Así como, se ha detallado el papel de LaaS en abordar los desafíos comunes y actuales en el desarrollo y la implementación de los laboratorios remotos así como la compartición inter-institucional, la interoperabilidad con otras aplicaciones heterogéneas, el acoplamiento con otros servicios heterogéneos y objetos educativos, la dificultad de desarrollo, y la estandarización.

Desde la perspectiva de bajo nivel, los futuros trabajos deben centrarse en expandir el rango de aplicaciones y en modularizar diferentes tipos de laboratorios remotos con diferentes escenarios de operación con fin de investigar más cuestiones y descubrir más soluciones y mejoras. La modularización implica primeramente sustituir las conexiones convencionales con conexiones basadas en estándares, así como añadir terminales de entrada y salida a cada componente de una manera abstracta con fin de permitir la máxima flexibilidad y manejabilidad.

Desde la perspectiva de alto nivel, los futuros trabajos deben centrarse en la estructura del repositorio Web de esos LaaSs y en la manera que los LaaSs se presentarán al público. Esto implica primariamente un mecanismo de reservas o de colas adecuado. Se adoptó el supuesto de "o bien el laboratorio está disponible o bien está ocupado" siendo la solución más simple que puede implementarse en varios modos. Por ejemplo, se puede comprobar si el laboratorio está disponible o si se está llevando a cabo en ese momento una sesión por otro usuario utilizando llamadas de servicios Web. Otro ejemplo, y el cual fue adoptado en el prototipo desarrollado en el Capítulo 6, sería no aceptar nuevas sesiones a no ser que ninguna sesión previa esté en funcionamiento. Además, el proveedor aún puede conceder acceso a ciertas direcciones públicas según convenios previos. Sin embargo, estas técnicas no serían suficientes para administrar una implementación de larga escala y con numerosos grupos de estudiantes. Aunque el enfoque mayor de esta tesis fue en el lado de nivel bajo, dejando el lado del nivel alto a los consumidores, uno puede argumentar que el proveedor está aún involucrado en cualquier transacción y por tanto, un mecanismo apropiado de reservas debe ser considerado desde el lado de nivel bajo e inicialmente considerado en los futuros diseños. Por lo tanto, el próximo desafío seria implementar un mecanismo de reservas utilizando capas extras y manteniendo, mientras tanto, los servicios expuestos lo más abstractos posibles, conforme a la premisa del paradigma LaaS.

B.7.1. Estandarización

El objetivo final del propuesto paradigma LaaS es establecer las bases de un modelo aceptable que pueda servir a los desarrolladores y a los proveedores de los laboratorios remotos. Para ello, LaaS fue desarrollado a base de un estudio amplio y un análisis de las ventajas y desventajas de las iniciativas y los esfuerzos previos de integración e implementación. Al mismo tiempo, se ha tenido en cuenta el contexto del futuro Web y los entornos educativos de próxima generación. Se ha propuesto un modelo inicial para el desarrollo de laboratorios remotos modulares y de acoplamiento débil que puedan implementarse con un alto nivel de abstracción y visualización. Esta propuesta fue descrita en el Capítulo 5 y clarificada mediante dos ejemplos teóricos. Seguidamente, se demostró su viabilidad y su posible implementación mediante un ejemplo práctico presentado en el Capítulo 6.

Sin embargo, no se puede definir un estándar a menos que haya un acuerdo global entre la mayoría de los miembros implicados en esta área. Una vez aceptada la propuesta o el concepto, se requeriría pocos pasos para conseguir una propuesta de estándar inicial. Las tareas principales deben centrarse en lo siguiente:

- Definir un formato estandarizado para exponer los servicios de los laboratorios.
- Ponerse de acuerdo sobre un metadatos estándar, para ser adoptado, e incluso sobre determinadas ontologías.
- Ponerse de acuerdo sobre algunos estándares industriales y de Web, para ser adoptados, como ODBC, IVI/VISA/ RESTful, y Websockets.

Esto se puede hacer posible con la ayuda de: el emergente grupo "*IEEE P1876™ Standard for Networked Smart Learning Objects for Online Laboratories*"; el consorcio GOLC (en inglés, Global Online Laboratory Consortium); y de otros socios de la industria involucrados o interesados en esta área como "National Instrumnets" y "Microsoft".

B.8. Resumen de los Resultados

Los resultados principales de esta tesis pueden resumirse y clasificarse por tres categorías principales como:

B.8.1. Investigación

Se ha desarrollado y justificado un nuevo paradigma, LaaS, para desarrollo e implementar laboratorios remotos modulares. Este paradigma permitirá superar todos los problemas actuales relacionados con el desarrollo y la implementación de los laboratorios remotos como la compartición inter-institucional, la interoperabilidad con otras aplicaciones heterogéneas, el acoplamiento con otros servicios heterogéneos y objetos educativos, la dificultad de desarrollo, y la estandarización. Para demostrar la viabilidad y la posible implementación de la solución propuesta, se ha desarrollado un prototipo y se ha examinado y se han obtenido resultados. (Capítulo 5 & 6)

Se han realizado los siguientes estudios como parte de la tarea de investigación:

1. Un estudio sobre las aplicaciones de los laboratorios remotos en todas las disciplinas de educación de ingeniería eléctrica. En este estudio se tratan los problemas de desarrollo e implementación, así como las limitaciones asociadas con diferentes tipos de aplicaciones (ej., aplicaciones de tiempo real, aplicaciones de alta tensión, etc.).
2. Un estudio sobre la integración de laboratorios remotos en diferentes tipos de sistemas educativos, enfatizando en las ventajas y desventajas de cada solución.
3. Un estudio sobre las características del presupuesto futuro Web, los entornos educativos de próxima generación, y los posibles métodos de acceso a los recursos educativos en línea en el futuro.
4. Un estudio sobre las tecnologías middleware disponibles para la implementación de una Arquitectura Orientada a Servicios.
5. Un estudio sobre la arquitectura genérica de los laboratorios remotos para todo tipo de aplicaciones y sobre sus componentes.
6. Un estudio sobre los conectores y los buses de comunicación estándares.
7. Un estudio sobre las tecnologías disponibles para desarrollar un software para el servidor de los laboratorios remotos.
8. Un estudio sobre las tecnologías de implementación de clientes Web e interfaces de usuario.

B.8.2. Desarrollo e implementación

Se ha desarrollado e implementado lo siguiente:

- Se ha desarrollado un conjunto de experimentos remotos de electrónica con VISIR. Este conjunto de experimentos está dirigido a los programas de grado de ingeniería, se ha implementado en programas oficiales de grado en la UNED y se han obtenido resultados de implementación. Además, este conjunto de experimentos ha sido implementado en el primer MOOC del mundo en electrónica basado en experimentación con laboratorios remotos, que fue proporcionado por DIEEC-UNED al público de todo el mundo. (Capítulo 3)

- Se ha desarrollado un conjunto de experimentos remotos de electrónica avanzada con VISIR. Este conjunto de experimentos está orientado a las necesidades de la industria y el mercado laboral y dirigido a los programas de posgrado y de aprendizaje con fin de reducir el hueco que hay entre el mundo académico y el mundo laboral. Este tipo de experimentos no se ha desarrollado previamente ni en laboratorios tradicionales ni en laboratorios remotos. Esta combinación de experimentos (los dos conjuntos montados), como resultado, convirtió VISIR en una plataforma única a nivel mundial de experimentos remotos de electrónica. Los experimentos han sido implementados en un máster Europeo inter-institucional a distancia en sistemas de comunicación e información y basado en experimentación con laboratorios remotos. Este programa de máster cuenta con 5 universidades Europeas cuyos miembros proceden de 4 diferentes países Europeos (entre ellos la UNED) y cada miembro participa con al menos una asignatura y con la aportación de un laboratorio remoto (en nuestro caso se proporcionó acceso a las prácticas desarrolladas dentro de la asignatura "*fuentes de alimentación para sistemas de tecnología de información y comunicación*" proporcionada por la UNED), lo cual convierte este programa de máster en un programa único en su tipo. (Capítulo 3)

- Se ha desarrollado el primer laboratorio remoto modular como un prototipo, y se han obtenido resultados de su utilización. (Capitulo 6)

B.8.3. Diseminación y Publicaciones[1]

- Primer autor de 4 artículos de revistas (i.e., las revistas incluyen *Industrial Electronics Magazine* y *IEEE Transactions on Learning Technologies*).
- Co-autor de 4 artículos de revistas (i.e., las revistas incluyen *IEEE Revista Iberoamericana electrónica de Tecnologías del Aprendizaje-RITA* y *Solar Energy*).
- Autor de un capítulo de libro orientado a la investigación, así como co-autor de otro capítulo de libro orientado a la investigación.
- Autor de 16 papers de conferencia y co-autor de otros 19 papers de conferencia (i.e., las conferencias incluyen: *Society for Information Technology & Teacher Education International Conference-SITE, ASEE Annual Conference & Exposition; IEEE Global Engineering Education Conference-EDUCON*; y *IEEE/ASEE Frontiers in Education Conference-FIE*A).
- Se han enviado más artículos y papers varios y actualmente están en revisión.

B.8.4. Otros Méritos[2]

- Se ha realizado una estancia de investigación de 6 meses en la Universidad de Tecnología, Sídney (UTS), Sídney, Australia.
- Se ha realizado una estancia de investigación de 5 meses en la Escuela Politécnica Federal de Lausana (EPFL), Lausana, Suiza.
- Se ha participado en 10 proyectos de investigación (i.e., 3 de ámbito nacional, 5 de ámbito Europeo, y 2 de ámbito internacional).

[1] La mayoría de las publicaciones fueron presentadas/publicadas en conferencias y revistas prestigiosas y de ámbito internacional.

[2] En la Sección 1.7, se presenta una lista detallada de los logros y contribuciones.

- Se obtuvo el premio del mejor paper "*Best Student Paper Award*" en la conferencia EDUCON 2012, Marrakech, Marrueco.
- Revisor en 3 revistas prestigiosas y de ámbito internacional (*International Journal of Engineering Education-IJEE*, *IEEE Network*, y *Computer Applications in Engineering Education)*, así como en 4 conferencias prestigiosas y de ámbito internacional (FIE 2012, FIE 2013, EDUCON 2012, y TALE 2012).

Appendix C

C.VISIR Configurations

In order to mount the new designed experiments discussed in Section 3-5, the following hardware and software configurations have been realized. Further information about the system installation can be consulted the "*VISIR Installation & Start-Up Guide*" [103], which is released within the official VISIR documents that are distributed among the SIG VISIR [42] mailing list (sig-visir@online-lists.org).

C.1. Hardware

Adding new circuits to VISIR is abide by the relay switching matrix capability which is described in Section 3.1.1. The matrix capability lies in its number of component boards (i.e., 16 boards maximum), number of relays on each board (i.e., 8 DPST + 2 DPST or 4SPST), and number of nodes propagating through all boards (i.e., A-I nodes + GND). Each two leads component (e.g. resistors, diodes, and capacitors) occupied a DPST relay. Each three leads component (e.g. transistors) occupied 3 SPST. The 4-pin bridge rectifier IC was connected to 4 SPST relays. For simplicity and space consideration, the shaded area of the converters' circuits shown in Figure. 13(d) and Figure 13(e) were mounted in an external circuit then connected to the matrix as a black box with input and output terminals as shown in Figure 15. Each terminal of the external circuits is connected to a SPST relay. Thus, users will only be concerned with the specific characteristics of the circuit rather than getting bogged down with its connections. Finally, necessary protection scheme for each circuit or component was realized as described in Section 3.5.

C.2. Software

The software configurations were realized as explained in the following steps:

- The “*Measurement Server*” software code (i.e., *component.types* file) was modified to add new types of component classes along with their properties: two-leads inductor (L), two-leads thermistors (NTC), 8-leads bridge rectifier IC (BR), and 9-leads converter (CONV) as follows:

```
# TypeConsFlags
# Flags:
#T can be turned around
#I Ignore value
#S Has special value
# use a newline at the end of every type

R                   2    T
L                   2    T
NTC                 2    T
C                   2    T
CE                  2
D                   2
VDCA                2    I    S
VDCB                2    I    S
VFGENA              2    I    S
VFGENB              2    I    S
VDC+6V              1    I    S
VDC+25V             1    I    S
VDC-25V             1    I    S
VDCCOM              1    I    S
5VA                 2    I
5VB                 2    I
SHORTCUT            2    T
IPROBE              2    T
DMM                 2
PROBE1              1
PROBE2              1
PROBE3              1
PROBE4              1
BLACKBOX            3
RELAY1C             3
XSWITCHOPEN         2    T    I
XSWITCHCLOSE        2    T    I
OP                  8
BR                  8
CONV                9
```

```
Q                   3
W                   2
POT                 3    S
# end of file
```

- Similarly, the new component types were listed in the "*equipment server*" software configuration file "*EquipmentServer.ini*" as follows:

```
[Global]
Port=5005 #Which port the server should listen on.
Reset Time = 300 [s]
Maximum Delay = 1000 [ms]
[Log]
Log Level = 4
Log File = C:\MeasuereServer.log
[Instrument Adress]
Function Generator 1 = PXI1Slot4
DC Power Supply 1 = PXI1Slot3
Digital Osciliscope 1 = PXI1Slot2
USB Interface 1 = USB0::0x1043::0x0000::NI-VISA-0::RAW
Digital Multimeter 1 = PXI1Slot5
[Digital Multimeter]
Powerline Frequency = 50 Hz #Not used anymore, but could be later on
:)
Timeout= 5000 [ms] #Not used anymore, but could be later on :)
[Ocilloscope]
Slave Trigger Delay = 0 #Removed, the two channel osc doesn't use it.
[Component type]
Component type =
D:2,C:2,VDCCOM:1,OP:8,BR:8,CONV:9,VFGENA:2,SHORTCUT:2,VDC25V:1,VDC-
25V:1,VDC6V:1,VFGENB:2,L:2,5VA:2,5VB:2,Q:3,BLACKBOX:3,
[Components list]
Components list = components.list
[Matrix version]
Matrix version = 4.1
```

- All the components used in the mounted circuits of the experiments were listed, with their corresponding connection and value, in the "*component list.list*" file of the "*equipment server*" software in order to be identified by it. The "*component list*" file (i.e., as well as the maxlist files) follows the PSpice netlist format. For instance, the switching converter can be listed as:

```
CONV_7_7:7_6:7_4:7_14:7_13:7_12:7_11  B H NC1 NC2 F E C G A  DC-DCSwitchingConverter5V
```

This code means that a component of class "*CONV*" is connected in the component board number 7 over the relays 7, 6, 4, 14, 13, 12, and 11 to the nodes B, H, F, E, C, G, and A, respectively. Since the converter class is of 9 pins and the switching converter only occupies 7 pins, the extra 2 pins are assigned *NC1* and *NC1*, which stand for "not connected". The term "*DC-DC Switching Converter 5V*" is what appears to users in the GUI. The current overall connection of the relay switching matrix is as follows:

```
VFGENA_24_1      A 0
VDC+25V_24_4:1_6 B
VDC-25V_24_5:1_7 C
VDC+6V_24_3:1_13 D
VDCCOM_24_2     0
SHORTCUT_1_1     I 0
SHORTCUT_1_2     A I
SHORTCUT_1_3     A H
SHORTCUT_1_5     A G
SHORTCUT_1_8     H 0
SHORTCUT_1_9     G 0
SHORTCUT_1_11    E 0
SHORTCUT_1_10    F 0
SHORTCUT_2_1     A F
SHORTCUT_2_2     A E
SHORTCUT_2_3     B I
SHORTCUT_2_5     B H
SHORTCUT_2_7     B G
SHORTCUT_2_8     B F
SHORTCUT_2_9     B E
SHORTCUT_2_10    E F
SHORTCUT_2_11    E G
SHORTCUT_2_13    E H
SHORTCUT_3_1     E I
SHORTCUT_3_2     F G
SHORTCUT_3_3     F H
SHORTCUT_3_5     F I
SHORTCUT_3_7     G H
SHORTCUT_3_8     G I
SHORTCUT_3_9     H I
R_3_10           H I 1
R_3_11           F G 1K
R_3_13           E F 2K
R_4_1            H I 10
R_4_2            H I 100
C_4_3            F G 100n
R_4_5            F G 3K
L_4_7            G H 10.1m
```

```
R_4_8            E F 511
R_4_9            E F 82.5
L_4_10           F G 9.9m
C_4_11           G H 1u
R_4_13           G H 100K
C_5_1            F G 47n
R_5_2            E F 47
D_5_3            E F 1N4007
C_5_5            F G 22u
R_5_7            A 0 4.7K
R_5_8            F G 511
R_5_9            F G 10K
D_5_10           E F BAT42
BR_5_13:5_14:5_12:5_11   A NC1 NC2 H G NC3 NC4 F  B80C1000
C_6_1            F G 470u
R_6_2            E F 1K
R_6_3            E F 470-3W
R_6_8            E F 75
NTC_6_10         H I NTC-10K-470-3W
CONV_6_7:6_6:6_5:7_5:6_4:6_14:6_13:6_12:6_11  B H D 0 F E C G A  DC-
DCLienarConverter5V
C_7_1            F G 10u
R_7_2            D 0 47K
R_7_3            B D 150K
R_7_8            C 0 470
NTC_7_9          B D NTC-10K-470-9W
R_7_10           B D 470-9W
CONV_7_7:7_6:7_4:7_14:7_13:7_12:7_11  B H NC1 NC2 F E C G A  DC-
DCSwitchingConverter5V
R_8_1            D E  1K
SHORTCUT_8_2     B D
C_8_3            F G  100n
C_8_5            F G  1u
C_8_7            F G  10u
R_8_8            E F  470
D_8_9            F G  Zener5v
D_8_10           F G  Zener5v2
R_8_11           A E  10K
R_8_13           D F  68K
R_9_1            D F  82K
R_9_2            D F  100K
R_9_3            D F  20K
C_9_8            E D  2n
R_9_9            H I  10K
SHORTCUT_9_10    E D
OP_9_7:9_6:9_5:9_4:9_14  NC1 D H C NC2 F B NC3  uA741
Q_9_13:9_12:9_11 F D C   2n2222
R_10_1           F 0  220
R_10_2           B C  820
R_10_3           B D  5.6K
C_10_8           A D  10u
```

```
C_10_9           C H  10u
R_10_10          D 0  10K
OP_10_7:10_6:10_5:10_4:10_14  NC1 D E C NC2 F B NC3  uA741
Q_10_13:10_12:1_14   C D B   2n2222
```

- For each exercise created by the teacher, a "*max list*" file is prepared in which the allowed connections or maximum values of instruments for such exercise are listed. For instance, a DC power supply can be listed as:

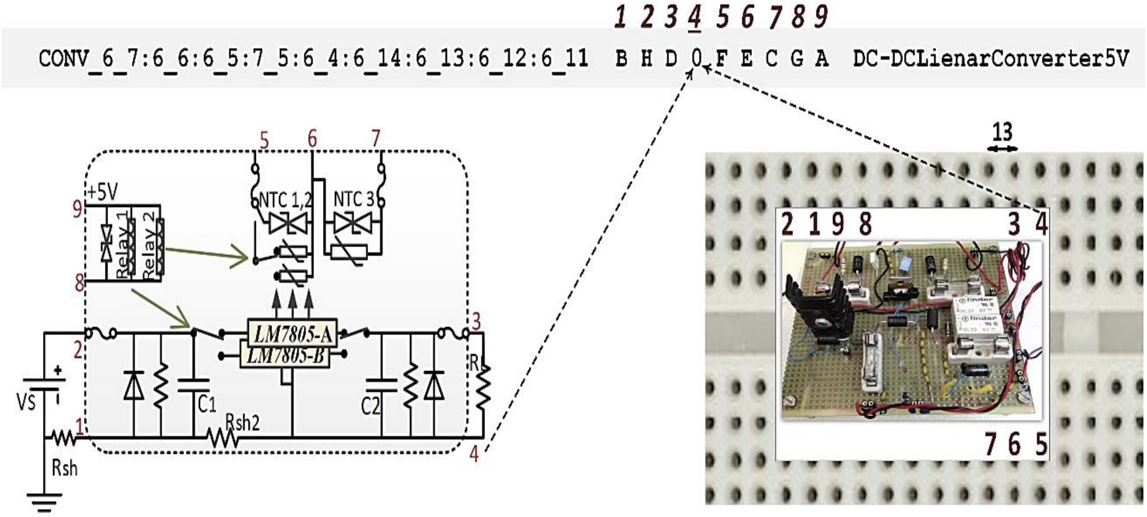
  ```
  VDC+25V_4B  max: 20 imax: 0.5
  ```

  This is to say that the maximum allowed values of the DC power supply—located in the "*source board*" and connected to the node B through the relay 4—are 20V and 0.5A, respectively.

- The last step is to modify the GUI package and add the new classes of components in an XML format, which gives information about the component's class, value, pins, position with respect to the PC-mouse cursor, possible rotations, and own photo. For instance, the linear converter was added as follows:

  ```
  <component type="CONV" value="DC-DCLinearConverter5V" pins="9">
    <rotations>
      <rotation ox="-65" oy ="-51.5" image="DC-DCLinearConverter5V.png"
      rot="0">
        <pins>
          <pin x="-65" y="-51.5"/>
          <pin x="-52" y="-51.5"/>
          <pin x="-39" y="-51.5"/>
          <pin x="-26" y="-51.5"/>
          <pin x="52"  y="-51.5"/>
          <pin x="65"  y="-51.5"/>
          <pin x="65"  y="51.5"/>
          <pin x="52"  y="51.5"/>
          <pin x="39"  y="51.5"/>
        </pins>
      </rotation>
    </rotations>
  </component>
  ```

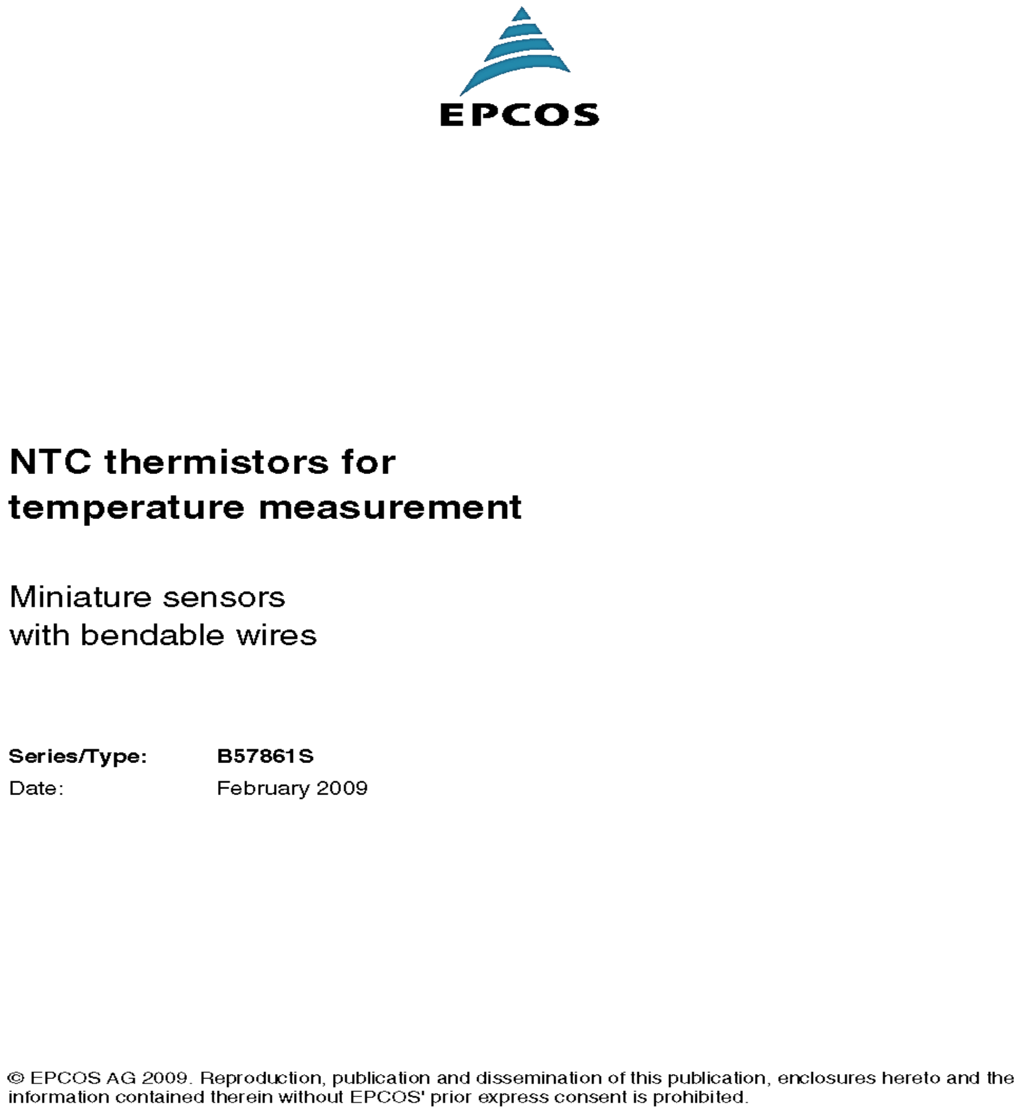
Notice that it has 9 pins with Cartesian coordinate with regarding to the PC-mouse pointer. The pins should coincide with the positions of the holes of the breadboard as shown in the above figure. The distance between two consecutive holes is 13 approximately.



```
CONV_6_7:6_6:6_5:7_5:6_4:6_14:6_13:6_12:6_11  B H D 0 F E C G A  DC-DCLienarConverter5V
```

This page intentionally left blank.

C.3. Datasheets



EPCOS

NTC thermistors for temperature measurement

Miniature sensors
with bendable wires

Series/Type:	**B57861S**
Date:	February 2009

R/T characteristics

R/T No.	**8016**		**8018**	
T (°C)	$B_{25/100}$ = 3988 K		$B_{25/100}$ = 3964 K	
	R_T/R_{25}	α (%/K)	R_T/R_{25}	α (%/K)
−55.0	96.3	7.4	–	–
−50.0	67.01	7.2	–	–
−45.0	47.17	6.9	–	–
−40.0	33.65	6.7	30.24	6.3
−35.0	24.26	6.4	22.1	6.1
−30.0	17.7	6.2	16.32	5.9
−25.0	13.04	6.0	12.17	5.8
−20.0	9.707	5.8	9.153	5.6
−15.0	7.293	5.6	6.945	5.4
−10.0	5.533	5.5	5.313	5.2
−5.0	4.232	5.3	4.097	5.1
0.0	3.265	5.1	3.183	4.9
5.0	2.539	5.0	2.491	4.8
10.0	1.99	4.8	1.963	4.7
15.0	1.571	4.7	1.557	4.6
20.0	1.249	4.5	1.244	4.4
25.0	1.0000	4.4	1.0000	4.3
30.0	0.8057	4.3	0.8083	4.2
35.0	0.6531	4.1	0.6572	4.1
40.0	0.5327	4.0	0.5373	4.0
45.0	0.4369	3.9	0.4418	3.9
50.0	0.3603	3.8	0.365	3.7
55.0	0.2986	3.7	0.303	3.7
60.0	0.2488	3.6	0.2527	3.6
65.0	0.2083	3.5	0.2118	3.5
70.0	0.1752	3.4	0.1783	3.4
75.0	0.1481	3.3	0.1508	3.3
80.0	0.1258	3.2	0.128	3.2
85.0	0.1072	3.2	0.1091	3.2
90.0	0.09177	3.1	0.0933	3.1
95.0	0.07885	3.0	0.08016	3.0
100.0	0.068	2.9	0.0691	2.9
105.0	0.05886	2.9	0.05974	2.9
110.0	0.05112	2.8	0.05183	2.8
115.0	0.04454	2.7	0.04512	2.8
120.0	0.03893	2.6	0.0394	2.7
125.0	0.03417	2.6	0.0345	2.6
130.0	0.03009	2.5	0.03032	2.6
135.0	0.02654	2.5	0.02672	2.5
140.0	0.02348	2.4	0.02361	2.5
145.0	0.02083	2.4	0.02091	2.4
150.0	0.01853	2.3	0.01857	2.4
155.0	0.01653	2.3	0.016537	2.3

FAIRCHILD
SEMICONDUCTOR®

August 2012

LM78XX/LM78XXA — 3-Terminal 1A Positive Voltage Regulator

LM78XX/LM78XXA
3-Terminal 1A Positive Voltage Regulator

Features

- Output Current up to 1A
- Output Voltages of 5, 6, 8, 9, 10, 12, 15, 18, 24
- Thermal Overload Protection
- Short Circuit Protection
- Output Transistor Safe Operating Area Protection

General Description

The LM78XX series of three terminal positive regulators are available in the TO-220 package and with several fixed output voltages, making them useful in a wide range of applications. Each type employs internal current limiting, thermal shut down and safe operating area protection, making it essentially indestructible. If adequate heat sinking is provided, they can deliver over 1A output current. Although designed primarily as fixed voltage regulators, these devices can be used with external components to obtain adjustable voltages and currents.

Ordering Information

Product Number	Output Voltage Tolerance	Package	Operating Temperature
LM7805CT	±4%	TO-220 (Single Gauge)	-40°C to +125°C
LM7806CT			
LM7808CT			
LM7809CT			
LM7810CT			
LM7812CT			
LM7815CT			
LM7818CT			
LM7824CT			
LM7805ACT	±2%		0°C to +125°C
LM7806ACT			
LM7808ACT			
LM7809ACT			
LM7810ACT			
LM7812ACT			
LM7815ACT			
LM7818ACT			
LM7824ACT			

Block Diagram

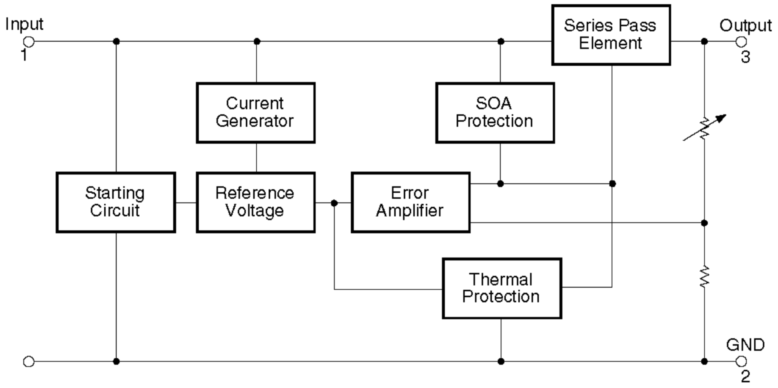


Figure 1.

Pin Assignment

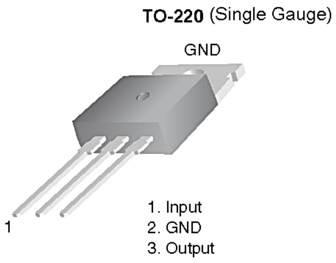


Figure 2.

Absolute Maximum Ratings

Absolute maximum ratings are those values beyond which damage to the device may occur. The datasheet specifications should be met, without exception, to ensure that the system design is reliable over its power supply, temperature, and output/input loading variables. Fairchild does not recommend operation outside datasheet specifications.

Symbol	Parameter		Value	Unit
V_I	Input Voltage	V_O = 5V to 18V	35	V
		V_O = 24V	40	V
$R_{\theta JC}$	Thermal Resistance Junction-Cases (TO-220)		5	°C/W
$R_{\theta JA}$	Thermal Resistance Junction-Air (TO-220)		65	°C/W
T_{OPR}	Operating Temperature Range	LM78xx	-40 to +125	°C
		LM78xxA	0 to +125	
T_{STG}	Storage Temperature Range		-65 to +150	°C

Features

- Efficiency up to 96%, no need for heatsinks!
- Pin-out compatible with LM78XX Linear Regs.
- Low profile (L*W*H=11.6*8.5*10.4mm)
- Wide input range (5V ~ 42V)
- Short circuit protection, thermal shutdown
- Non standard outputs available as specials
- Low ripple and noise
- See Innoline App Notes for use as a positive-to-negative inverter (alternative to 79xx regulator)

INNOLINE
DC/DC-Converter
with 3 year Warranty

RECOM

1.0 AMP
SIP3
Single Output

Selection Guide

Part Number SIP3	Input Range (V)	Output Voltage (V)	Output Current (A)	Efficiency Min. Vin (%)	 Max. Vin (%)
R-78C1.8-1.0	5 – 42	1.8	1.0	80	71
R-78C3.3-1.0	7 – 42	3.3	1.0	89	79
R-78C5.0-1.0	8 – 42	5	1.0	93	85
R-78C9.0-1.0	12 – 42	9	1.0	95	90
R-78C12-1.0	15 – 42	12	1.0	96	92
R-78C15-1.0	18 – 42	15	1.0	96	94

RoHS
2002/95/EC
6/6

EN-60950-1 Certified

Specifications (typical at 25°C, 10% minimum load, unless otherwise specified)

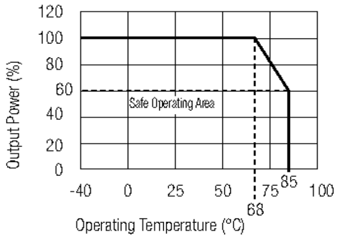
Characteristics	Conditions	Min.	Typ.	Max.
Input Voltage Range	All Series	Vout+3V		42V
Output Voltage Range	All Series	1.8V		15V
Output Current	All Series	0mA*		1000mA
Output Current Limit	All Series			3000mA
Short Circuit Input Current (Vin =24V)	All Series		65mA	
No Load Input Current			1mA	
Short Circuit Protection		Continuous, automatic recovery		
Output Voltage Accuracy (At 100% Load)	All Series		±2%	±3%
Line Regulation (100% Load, Vin max.)	All Series		0.2%	
Load Regulation (10 to 100% full load)	All Series		0.4%	
Dynamic Load Stability	100% <-> 50% load			±75mV
	100% <-> 10% load			±200mV
Ripple & Noise (20Mhz BW Limited)	Vin = 24V, Vout =1.8V-15V		75mVp-p	100mVp-p
With 10µF MLCC output capacitor	Full Load		30mVp-p	
Temperature Coefficient	-40°C ~ +85°C ambient			0.015%/°C
Max capacitance Load	with normal start-up time, no external components			470µF
	with <1 second start up time + diode protection circuit			6800µF
Switching Frequency		280kHz	350kHz	420kHz
Operating Temperature Range		-40°C		+85°C
Maximum Case Temperature				+100°C
Storage Temperature Range		-55°C		+125°C
Case Thermal Impendance				70°C/W
Thermal Shutdown	Internal IC junction			+160°C
Conducted Emissions (with filter)	EN55022			Class B
Radiated Emissions (with filter)	EN55022			Class B
ESD	EN61000-4-2			Class A
Radiated Immunity	EN61000-4-3			Class A
Fast Transient	EN61000-4-4			Class A
Conducted Immunity	EN61000-4-6			Class A
Magnetic Field Immunity	EN61000-4-8			Class A
Package Weight				2g
Packing Quantity				42 pcs per Tube

R-78C-1.0

Description

The R-78Cxx-1.0 series switching regulators are ideally suited to replace 1 Amp 78xx linear regulators and are pin compatible. Efficiencies of up to 96% means that very little energy is wasted as heat and the high input voltage is a useful feature.

Derating-Graph
(Ambient Temperature)



This page intentionally left blank.

Appendix D

D. ReLaSIS

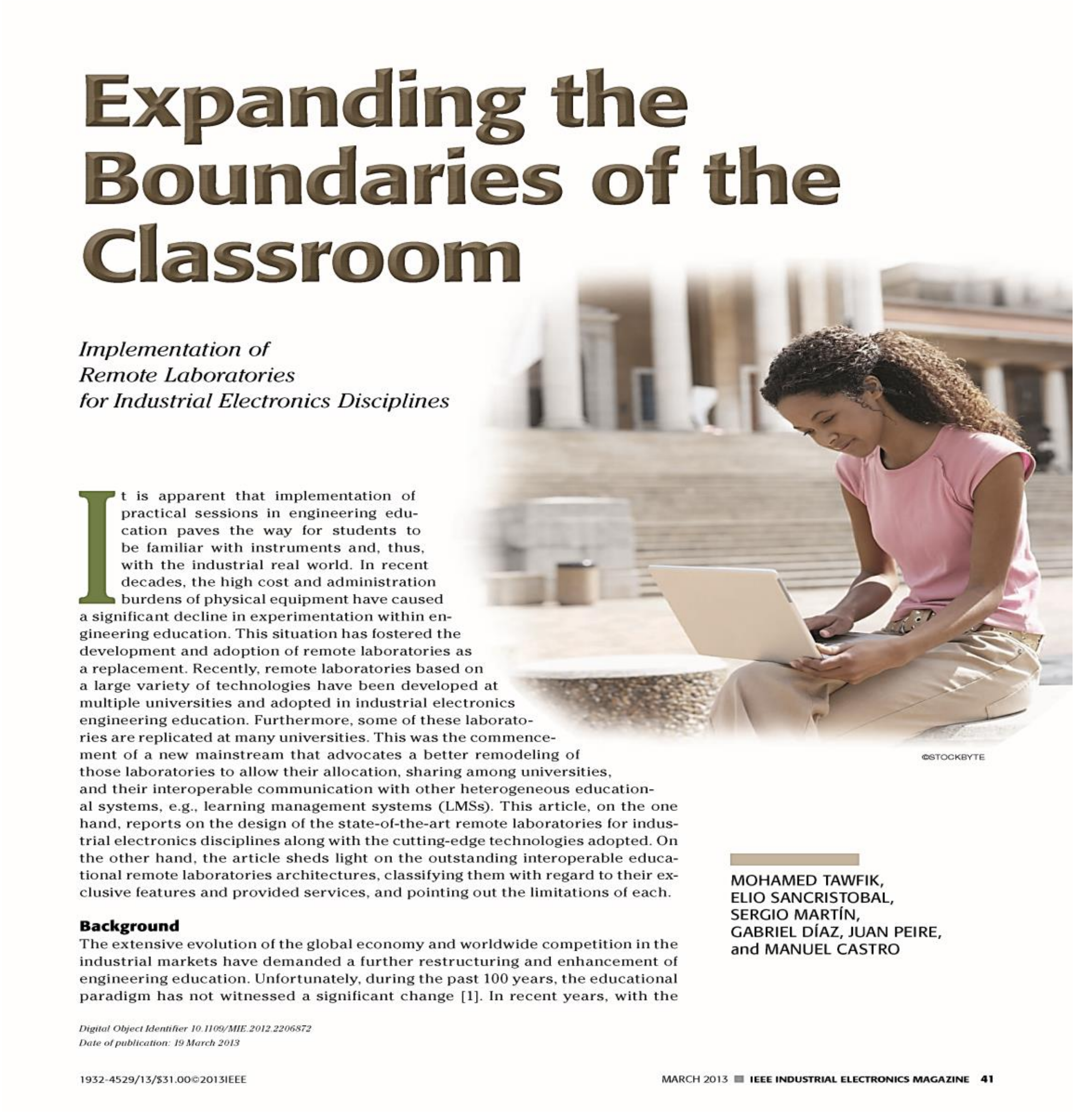
This specification was developed by Prof. David Lowe, the current president of GOLC, in collaboration with other members within GOLC. The current version (V0.8c) was released in December 5, 2011 and was termed "*Remote Laboratory Systems Interoperability Standard: Interface Definitions*". It is composed of 6 base-line profiles and each profile supports a collection of calls as shown in the following tables.

Profile	**Description**
System Query profile	Supports basic system enquiries required to determine agreed supported services between consumers and providers (the baseline principles)
Rig Query profile	Supports basic system enquiries required to determine the rig sets that are made available by a lab provider
Experiment Query profile:	Supports basic system enquiries required to determine the executed experiments that have been carried out by a lab provider.
Basic Rig Access profile	This profile provides support for basic access to provided rig sets or specific rigs. Two primary modes of access are (optionally) supported by the provider: queuing, where the user can request access and they then enter a queue with access being granted when they reach the front of the queue and a rig becomes available; and booking, where the user can make a booking for a particular time slot
Basic Batch Experiment profile	Basic support for execution of batch based laboratories using either provider–supplied or consumer-supplied experiment interface
Basic Interactive Experiment profile	Basic support for execution of interactive based laboratories using either provider –supplied or consumer-supplied experiment interface

System Query	**QueryProfiles()**	Obtains a list of profiles (and profile versions) supported by the provider.
	QueryServiceLevel()	Obtains provider-specific information on the level of his provided service.
Rig Query profile	**GetRigs()**	If no rig sets specified, it returns a list of unique top level rig set laboratory identifiers that are accessible to this consumer. If a rig set is specified, it returns the children rig sets.
	GetRigStatus()	Provides the current status of the rig set, and whether it is currently operational.
	GetRigInfo()	Obtains any relevant property information pertaining to the rig set.
Experiment Query profile	**GetExperimentIDs()**	Retrieves a list of experiment IDs that are active and associated with rigs in this rig set, and related to a particular consumer, consumer group or user.
	GetExperimentStatus()	Indicates the current status of an experiment, whether the experiment is running (and if so then for how long), or if it is in an idle state.
	GetExperimentResults()	Retrieves the results from a (partially or fully) completed experiment execution.
Basic Rig Access profile	**RigAccessType()**	Determines whether a particular rig set is bookable or queuable (or both).
	RigAvailability()	For bookable rig sets, it allows querying the time periods within a given timeframe that are currently available for booking.
	RigBooking()	Requests a booking to be made for the given experiment definition. An email address can be associated with the booking for confirmation.
	RigBookingCancel()	Cancels an existing booking.
	RigBookingStatus()	Queries the status of an existing booking and in particular if the booking is currently available for redemption.
	RigBookingsQuery()	Retrieves a list of all existing bookings that have been made for the specified consumer, user group, or user.
	RigQueue()	Adds the given experiment definition into the queue.
	RigQueueCancel()	Removes a given experiment definition from a queue.
	RigQueueStatus()	Requests information on the current status of an experiment definition with a queue.
Basic Batch Experiment profile	**BatchExpCreate()**	Creates a blank "empty" experiment definition on the provider, to be executed on the permitted specified rig set.
	BatchExpDelete ()	Deletes a previously created experiment definition.
	BatchExpGetSpecs()	Obtains a list of the information required for successful execution of an experiment.
	BatchExpSetDefn()	Provides the information required to define a valid experiment and saves this into an existing experiment.
	BatchExpGetDefn()	Retrieves the experiment definition from a previously defined experiment.
	BatchExpExecute()	Requests the execution of a previously defined batch experiment. The experiment (if accepted for execution) will then be queued for execution.
	BatchExpLaunchProvIF()	Provides a URL to use in launching a user interface associated with the specified experiment.
Basic Interactive Experiment	**InteractiveExpCreate()**	Creates a blank "empty" experiment definition on the provider, to be executed on the permitted rig set.
	InteractiveExpDelete()	Deletes a previously created experiment definition.
	InteractiveExpLaunchConsIF()	Requests the commencement of an interactive experiment using a consumer provided interface.
	InteractiveExpLaunchProvIF()	Provides a URL to use in launching a user interface associated with the specified experiment.

Appendix E

E. Relevant Merits



Expanding the Boundaries of the Classroom

Implementation of Remote Laboratories for Industrial Electronics Disciplines

It is apparent that implementation of practical sessions in engineering education paves the way for students to be familiar with instruments and, thus, with the industrial real world. In recent decades, the high cost and administration burdens of physical equipment have caused a significant decline in experimentation within engineering education. This situation has fostered the development and adoption of remote laboratories as a replacement. Recently, remote laboratories based on a large variety of technologies have been developed at multiple universities and adopted in industrial electronics engineering education. Furthermore, some of these laboratories are replicated at many universities. This was the commencement of a new mainstream that advocates a better remodeling of those laboratories to allow their allocation, sharing among universities, and their interoperable communication with other heterogeneous educational systems, e.g., learning management systems (LMSs). This article, on the one hand, reports on the design of the state-of-the-art remote laboratories for industrial electronics disciplines along with the cutting-edge technologies adopted. On the other hand, the article sheds light on the outstanding interoperable educational remote laboratories architectures, classifying them with regard to their exclusive features and provided services, and pointing out the limitations of each.

©STOCKBYTE

MOHAMED TAWFIK,
ELIO SANCRISTOBAL,
SERGIO MARTÍN,
GABRIEL DÍAZ, JUAN PEIRE,
and MANUEL CASTRO

Background

The extensive evolution of the global economy and worldwide competition in the industrial markets have demanded a further restructuring and enhancement of engineering education. Unfortunately, during the past 100 years, the educational paradigm has not witnessed a significant change [1]. In recent years, with the

Digital Object Identifier 10.1109/MIE.2012.2206872
Date of publication: 19 March 2013

1932-4529/13/$31.00©2013IEEE

Virtual Instrument Systems in Reality (VISIR) for Remote Wiring and Measurement of Electronic Circuits on Breadboard

Mohamed Tawfik, *Member*, *IEEE*, Elio Sancristobal, *Member*, *IEEE*, Sergio Martin, *Member*, *IEEE*, Rosario Gil, *Member*, *IEEE*, Gabriel Diaz, *Senior Member*, *IEEE*, Antonio Colmenar, Juan Peire, *Senior Member*, *IEEE*, Manuel Castro, *Fellow*, *IEEE*, Kristian Nilsson, Johan Zackrisson, Lars Håkansson, and Ingvar Gustavsson, *Member*, *IEEE*

Abstract—This paper reports on a state-of-the-art remote laboratory project called Virtual Instrument Systems in Reality (VISIR). VISIR allows wiring and measuring of electronic circuits remotely on a virtual workbench that replicates physical circuit breadboards. The wiring mechanism is developed by means of a relay switching matrix connected to a PCI eXtensions for Instrumentation (PXI) instrumentation platform. The entire equipment is controlled by LabVIEW server software, in addition to a measurement server software that protects the equipment from hazard connections by verifying input circuit designs, sent by students, before being executed. This paper addresses other approaches such as remote labs based on Data Acquisition Cards (DAQs), NetLab, and RemotElectLab, comparing them with VISIR in order to emphasize its singularity. Topics discussed are as follows: the technical description, software, operation cycle, features, and provided services. In addition, the feedback received by students at several universities and the encountered drawbacks along with the proposed solutions are highlighted. The paper finally addresses the ongoing and future challenges within the VISIR community including its integration with Learning Management Systems (LMSs) and iLab Shared Architecture (ISA), its new hardware version release that is based on LAN eXtensions for Instrumentation (LXI), and its new open platform version that supports federated access.

Index Terms—Computer uses in education, computer-aided engineering, electronics, emerging technologies

1 Introduction

The Signal Processing Department (ASB) at Blekinge Institute of Technology (BTH) in Sweden together with National Instruments in the USA (as a supplier of instruments) and Axiom EduTECH in Sweden (as a supplier of education, technical software, and engineering services for noise and vibration analysis) launched the Virtual Instrument Systems in Reality (VISIR) Project (http://openlabs.bth.se/electronics) at the end of 2006. The project was financially supported by BTH and the Swedish Governmental Agency for Innovation Systems (VINNOVA).VISIR is a remote laboratory [1] for wiring and measuring electronics circuits on a breadboard remotely. The user designs and constructs her circuit via PC-mouse on a seamlessly simulated workbench that resembles the real lab elements and components. Once the designed circuit is submitted, it is first sent to be verified, then it is sent to be wired and measured by real instruments, and finally, it is received by the user on her PC-screen in real time [2]. VISIR allows realizing real measurement on physical equipment, which is not possible to achieve using simulation software. For instance, the resistance of a 10k Ω resistor may vary due to many factors such as temperature and hours of utilization. Thus, its real value measured by VISIR is not the same as its simulated value measured by an electronic circuit simulation software such as NI Multisim (www.ni. com/multisim/); the value measured by VISIR is 9.958k Ω, while the value obtained by NI Multisimis 10k Ω, as demonstrated in Fig. 1.

So far, six universities have already implemented VISIR after Blekinge Institute of Technology: Carinthia University of Applied Sciences and FH Campus Wien for Applied Sciences, both in Austria; Polytechnic Institute of Porto (ISEP) in Portugal; University of Deusto and the Spanish University for Distance Education (UNED), both in Spain; and Madras Institute of Technology (IIT-M) in India. The following universities have shown their interest in participating in this project but they have not yet implemented it: University of Genoa in Italy, Princess Sumaya University for Technology in Jordan, Gunadarma University in Indonesia, Institute for the Development of New Technologies (UNINOVA) in Portugal, and College of the North Atlantica in Qatar. A Special Interest Group of VISIR (SIG VISIR) was created by the International Association of

- *M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, A. Colmenar, J. Peire, and M. Castro are with the Departamento de Eléctrica, Electrónica y Control Industrial (DIEEC), Escuela Técnica Superior de Ingenieros Industriales (ETSII), Universidad Nacional de Educación a Distancia (UNED), Calle Juan del Rosal, 12, Madrid 28040, Spain. E-mail: {mtawfik, elio, smartin, rgil, gdiaz, acolmenar, jpeire, mcastro}@ieec.uned.es.*
- *K. Nilsson, J. Zackrisson, L. Håkansson, and I. Gustavsson are with ING-AET, Blekinge Institute of Technology, SE-371 79 Karlskrona, Sweden. E-mail: {kristian.nilsson, johan.zackrisson, lars.hakansson, ingvar.gustavsson}@bth.se.*

Manuscript received 31 July 2011; revised 10 Mar. 2012; accepted 16 Aug. 2012; published online 17 Sept. 2012.
For information on obtaining reprints of this article, please send e-mail to: lt@computer.org, and reference IEEECS Log Number TLT-2011-07-0083.
Digital Object Identifier no. 10.1109/TLT.2012.20.

On the Design of Remote Laboratories

A Study on the Lab Server Software Technologies

Mohamed Tawfik, Elio Sancristobal, Sergio Martin, Rosario Gil, Gabriel Diaz, Juan Peire, Manuel Castro
Electrical and Computer Engineering Department
Spanish University for Distance Education
Madrid, Spain
{mtawfik, elio, smartin, rgil, gdiaz, jpeire, mcastro}@ieec.uned.es

***Abstract*—This paper presents a thorough study on the available lab server software technologies for the design of remote laboratories. The paper focuses on the common outstanding technologies applied for this purpose such as LabVIEW and MATLAB. The paper addresses as well on other solutions such as high-level programming languages, e.g., C/C++ and .NET languages, pointing out the advantages, limitations, and applications of each. This is in order to ease the selection of the appropriate software when designing a remote lab. Furthermore, the paper points out applications in which various technologies are adopted in order to make use of the exclusive features of each. The study is preceded by a brief overview on the main hardware and software components in the construction of today's remote laboratory for engineering practice to emphasize the role of the controller server in the lab operation.**

Keywords-component; distance education; e-learning; on-line learning; Remote Laboratoy

I. Introduction

Over the last decade, and with the evolving of computer and communication technologies, we have witnessed significant breakthroughs in the development of remote laboratories and in their adoption in engineering education. The paradigm of this adoption is more likely to rise and the way is irreversible. This is owing to the role of remote laboratories in removing the administration and cost burdens that hinder the adoption of hands-on lab sessions in engineering education.

In the literature [1]-[3], several studies have been carried out on dozens of remote laboratories, in multiple disciplines, that are installed at several universities around the world. This could not only justify that remote laboratories development and adoption have reached its maturity, but also, it could prove that there exit several solutions in a competitive market. For instance, in [1], state-of-art remote laboratories are listed according to their disciplines, stating the supporting technologies and current trends. In [2], a through literature review has been carried out on 42 remote laboratories developed in the past ten years. The review encompasses their classification per scientific field, programming languages, and communication tools. In [3], a remarkable study has been realized on the client software technologies for building a remote laboratory, providing effective comparisons and opinions of experts. This paper sheds the light on the lab server software technologies such as LabVIEW (Laboratory Virtual Instrumentation Engineering Workbench), MATLAB (Matrix Laboratory), and high-level programming languages. The paper points out the exclusive features and limitations of each, with applications examples, in order to ease the selection of the appropriate software when designing a remote lab.

II. Common Remote Laboratory Architecture

Remote laboratories are those laboratories that can be controlled and administrated online. They differ from the virtual simulated laboratories as they are interacting with physical instruments. The common generic architecture design of today's remote laboratory could be structured as shown in Fig.1. Next, a comprehensive definition of the main components of this architecture is going to be presented, pointing out the available technologies they mostly rely on.

A. *User Interface*

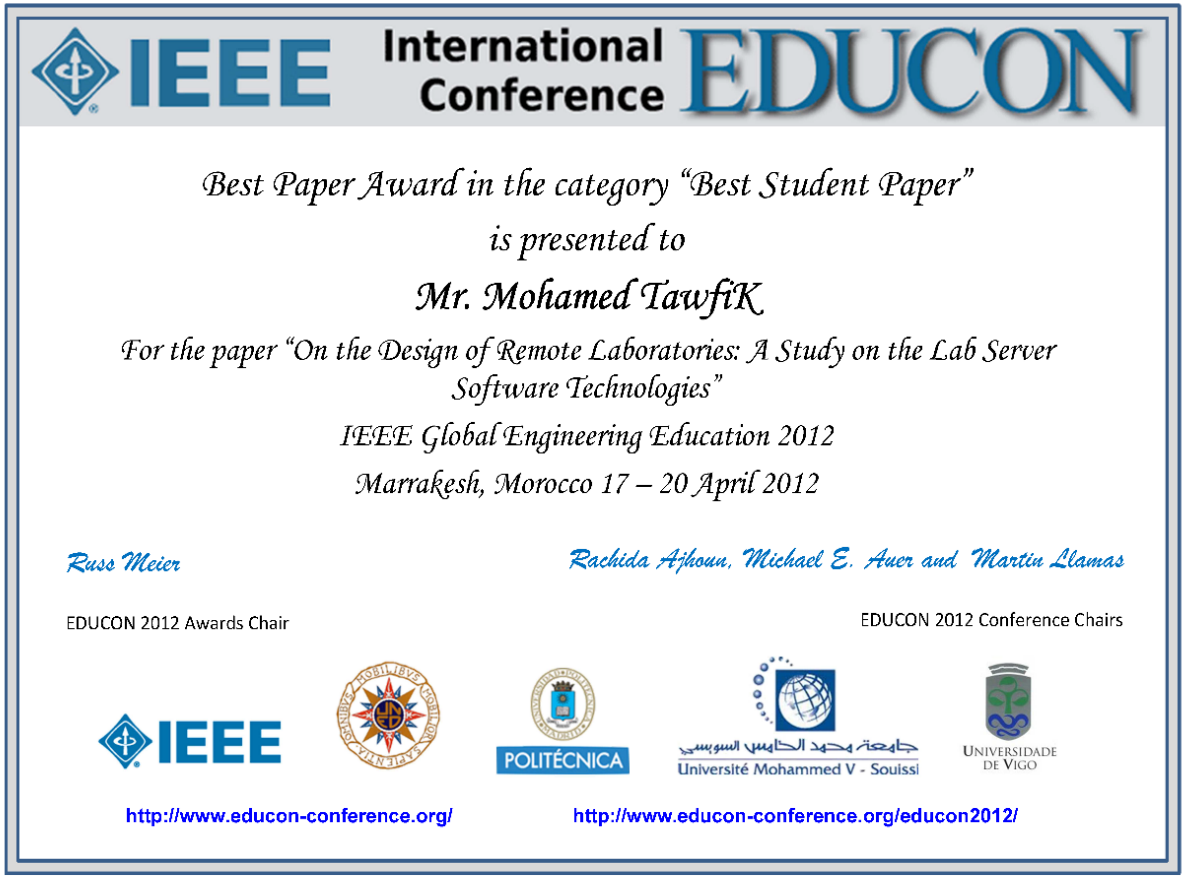
The user interface is a virtual end-user workbench that handles all the lab administration process. It is a web site that runs on the user's web browser and usually requires a server-side programming language to retrieve user's data from database such as PHP, ASP, and JSP, along with a Graphical User Interface (GUI), which is built by an animation technology embedded in the HTML code to resemble the real laboratory workbench such as Flash, Java Applets and MS Silverlight. Other scripting languages commonly used are: JavaScript; it adds interactive elements to the web page without relying on the server, and AJAX; it retrieves information from the server in an efficient way and without needing to refresh the web page. The website could be supplanted by a software application written in JAVA, .NET or C/C++ languages which is installed on the user-PC and connected to a database server (application server).

B. *Web Server*

The web server is a server-PC that hosts the web site and the database files. Apache and Microsoft IIS are the most commonly used servers, while MySQL, Microsoft SQL, and Oracle, are the most commonly used databases. The web server sends the user requests to the lab server in the form of XML messages through TCP/IP model over HTTP layer. Other attempts have been made to use non-HTTP based protocols such as Common Objective Request Broker Architecture (CORBA), Java Remote Method Invocation (RMI), .NET Remoting, and TCP/IP sockets. Some of these technologies, however, are restricted to local networks only [3].

This research is sponsored by the project s-Labs (TIN2008- 06083 -C03-01/TSI) from the Spanish Ministry for Science and Innovation.

978-1-4673-1456-5/12/$31.00 ©2012 IEEE



IEEE International Conference EDUCON

Best Paper Award in the category "Best Student Paper"

is presented to

Mr. Mohamed TawfiK

For the paper "On the Design of Remote Laboratories: A Study on the Lab Server Software Technologies"

IEEE Global Engineering Education 2012

Marrakesh, Morocco 17 – 20 April 2012

Russ Meier

EDUCON 2012 Awards Chair

Rachida Ajhoun, Michael E. Auer and Martin Llamas

EDUCON 2012 Conference Chairs

IEEE

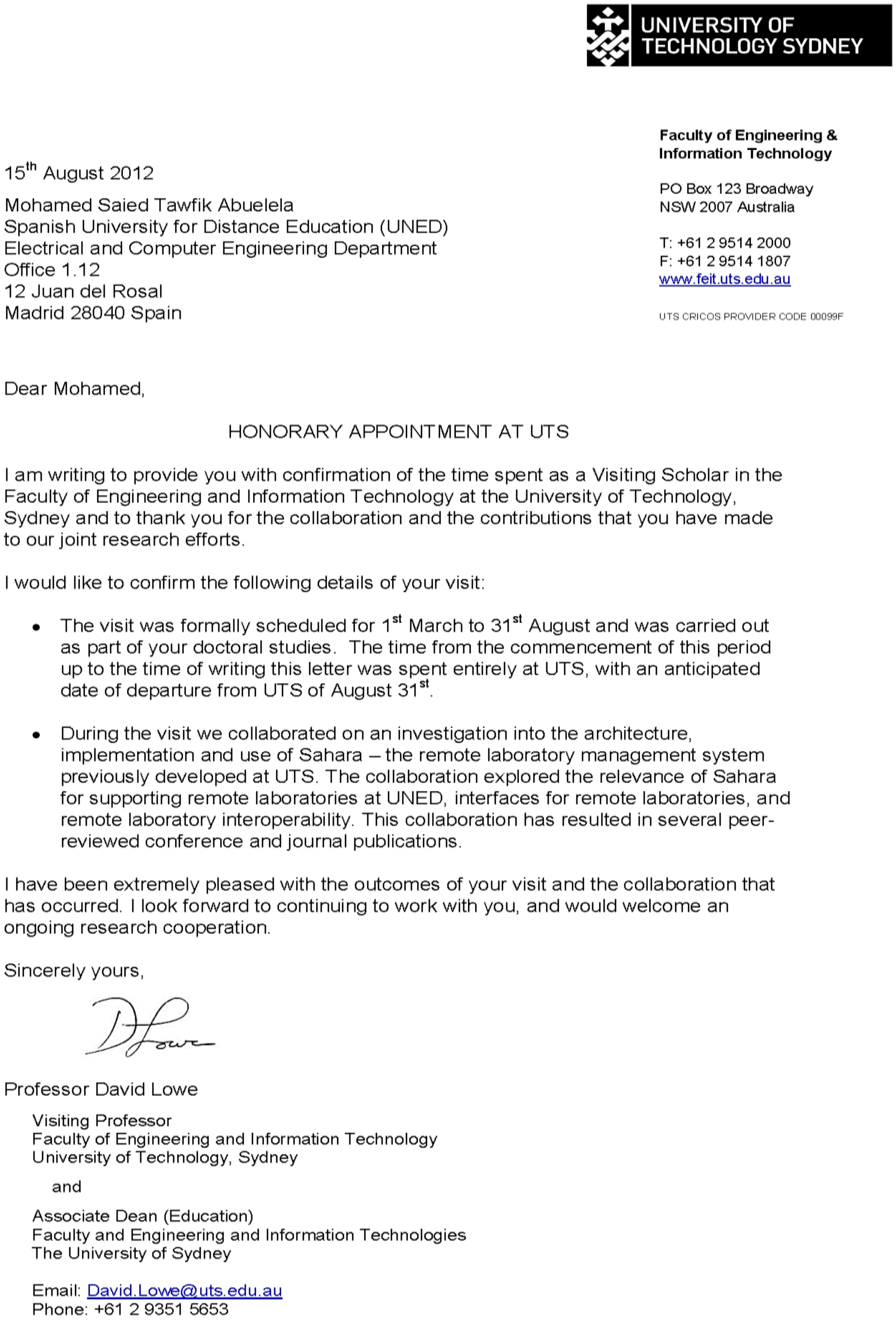
POLITÉCNICA

جامعة محمد الخامس السويسي
Université Mohammed V - Souissi

UNIVERSIDADE DE VIGO

http://www.educon-conference.org/

http://www.educon-conference.org/educon2012/



UNIVERSITY OF
TECHNOLOGY SYDNEY

Faculty of Engineering &
Information Technology

PO Box 123 Broadway
NSW 2007 Australia

T: +61 2 9514 2000
F: +61 2 9514 1807
www.feit.uts.edu.au

UTS CRICOS PROVIDER CODE 00099F

15th August 2012

Mohamed Saied Tawfik Abuelela
Spanish University for Distance Education (UNED)
Electrical and Computer Engineering Department
Office 1.12
12 Juan del Rosal
Madrid 28040 Spain

Dear Mohamed,

HONORARY APPOINTMENT AT UTS

I am writing to provide you with confirmation of the time spent as a Visiting Scholar in the Faculty of Engineering and Information Technology at the University of Technology, Sydney and to thank you for the collaboration and the contributions that you have made to our joint research efforts.

I would like to confirm the following details of your visit:

- The visit was formally scheduled for 1st March to 31st August and was carried out as part of your doctoral studies. The time from the commencement of this period up to the time of writing this letter was spent entirely at UTS, with an anticipated date of departure from UTS of August 31st.

- During the visit we collaborated on an investigation into the architecture, implementation and use of Sahara – the remote laboratory management system previously developed at UTS. The collaboration explored the relevance of Sahara for supporting remote laboratories at UNED, interfaces for remote laboratories, and remote laboratory interoperability. This collaboration has resulted in several peer-reviewed conference and journal publications.

I have been extremely pleased with the outcomes of your visit and the collaboration that has occurred. I look forward to continuing to work with you, and would welcome an ongoing research cooperation.

Sincerely yours,

DLowe

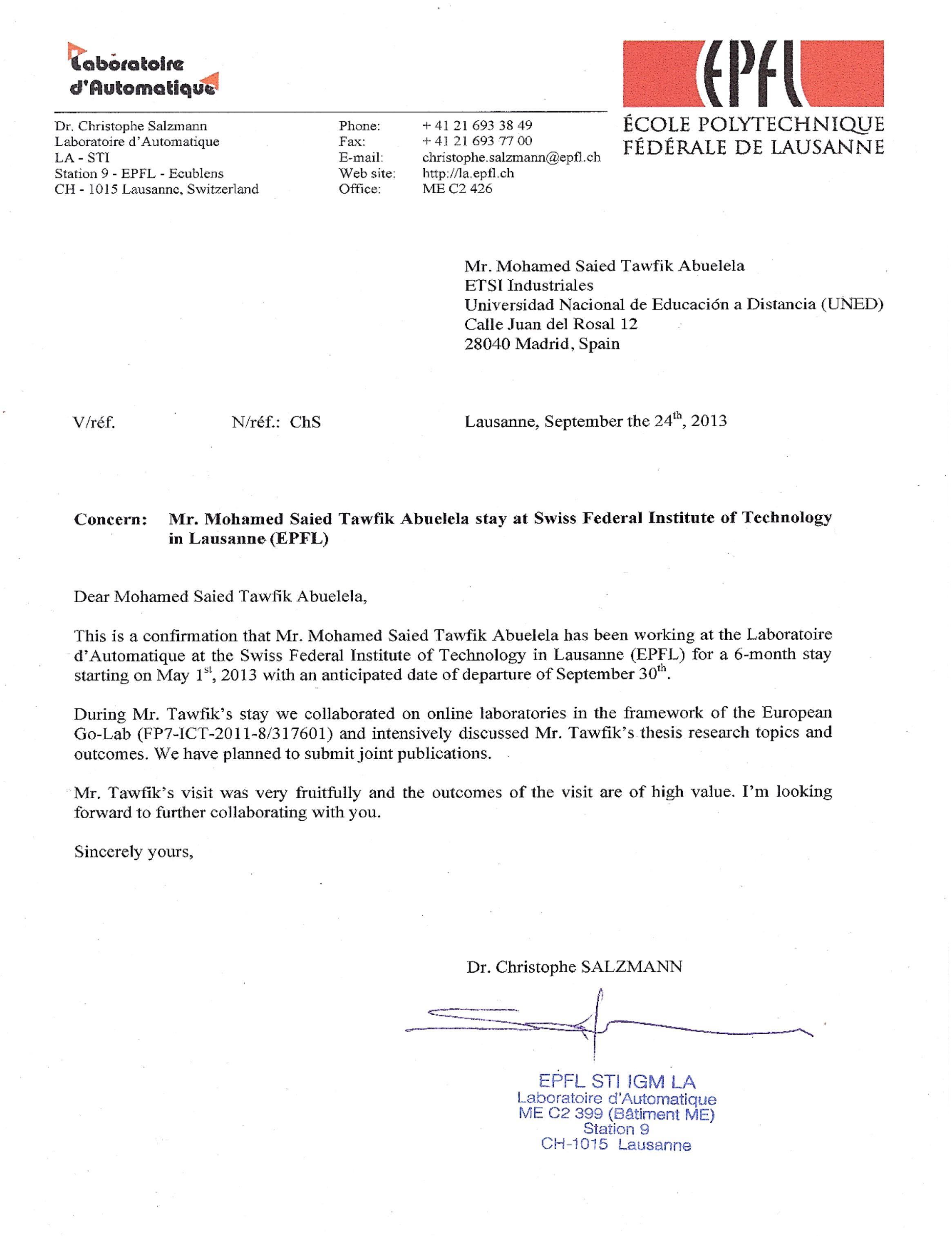
Professor David Lowe

Visiting Professor
Faculty of Engineering and Information Technology
University of Technology, Sydney

and

Associate Dean (Education)
Faculty and Engineering and Information Technologies
The University of Sydney

Email: David.Lowe@uts.edu.au
Phone: +61 2 9351 5653



Laboratoire
d'Automatique

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Dr. Christophe Salzmann
Laboratoire d'Automatique
LA - STI
Station 9 - EPFL - Ecublens
CH - 1015 Lausanne, Switzerland

Phone: + 41 21 693 38 49
Fax: + 41 21 693 77 00
E-mail: christophe.salzmann@epfl.ch
Web site: http://la.epfl.ch
Office: ME C2 426

Mr. Mohamed Saied Tawfik Abuelela
ETSI Industriales
Universidad Nacional de Educación a Distancia (UNED)
Calle Juan del Rosal 12
28040 Madrid, Spain

V/réf. N/réf.: ChS Lausanne, September the 24th, 2013

Concern: Mr. Mohamed Saied Tawfik Abuelela stay at Swiss Federal Institute of Technology in Lausanne (EPFL)

Dear Mohamed Saied Tawfik Abuelela,

This is a confirmation that Mr. Mohamed Saied Tawfik Abuelela has been working at the Laboratoire d'Automatique at the Swiss Federal Institute of Technology in Lausanne (EPFL) for a 6-month stay starting on May 1st, 2013 with an anticipated date of departure of September 30th.

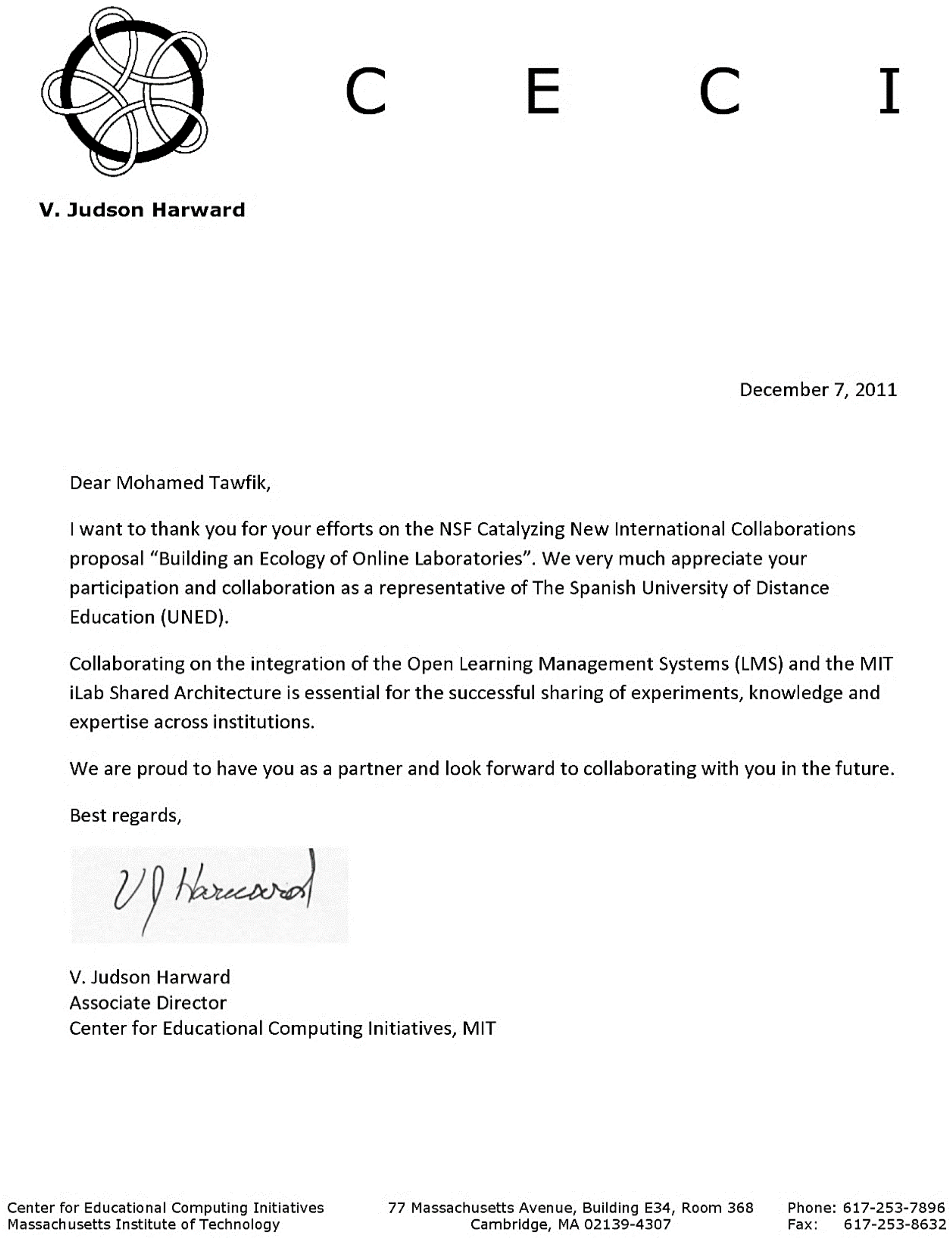
During Mr. Tawfik's stay we collaborated on online laboratories in the framework of the European Go-Lab (FP7-ICT-2011-8/317601) and intensively discussed Mr. Tawfik's thesis research topics and outcomes. We have planned to submit joint publications.

Mr. Tawfik's visit was very fruitfully and the outcomes of the visit are of high value. I'm looking forward to further collaborating with you.

Sincerely yours,

Dr. Christophe SALZMANN

EPFL STI IGM LA
Laboratoire d'Automatique
ME C2 399 (Bâtiment ME)
Station 9
CH-1015 Lausanne



C E C I

V. Judson Harward

December 7, 2011

Dear Mohamed Tawfik,

I want to thank you for your efforts on the NSF Catalyzing New International Collaborations proposal “Building an Ecology of Online Laboratories”. We very much appreciate your participation and collaboration as a representative of The Spanish University of Distance Education (UNED).

Collaborating on the integration of the Open Learning Management Systems (LMS) and the MIT iLab Shared Architecture is essential for the successful sharing of experiments, knowledge and expertise across institutions.

We are proud to have you as a partner and look forward to collaborating with you in the future.

Best regards,

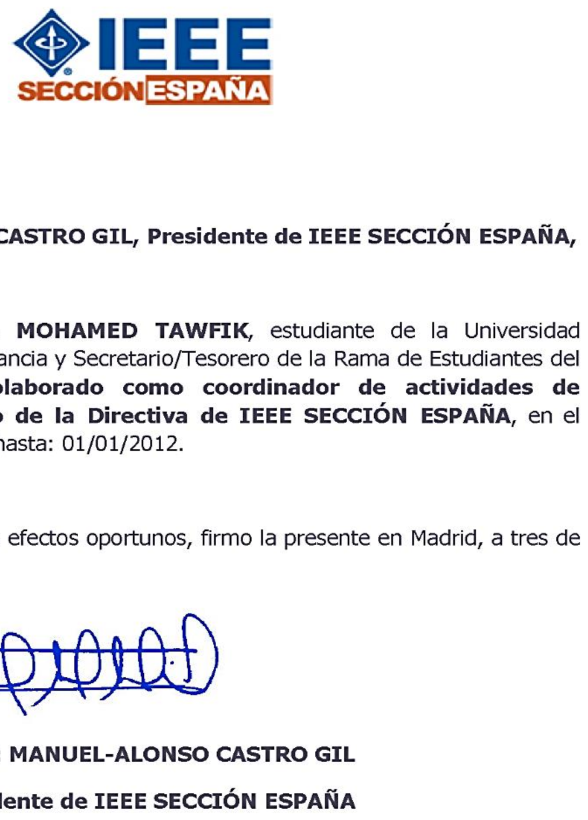
VJ Harward

V. Judson Harward
Associate Director
Center for Educational Computing Initiatives, MIT

Center for Educational Computing Initiatives
Massachusetts Institute of Technology

77 Massachusetts Avenue, Building E34, Room 368
Cambridge, MA 02139-4307

Phone: 617-253-7896
Fax: 617-253-8632



IEEE SECCIÓN ESPAÑA

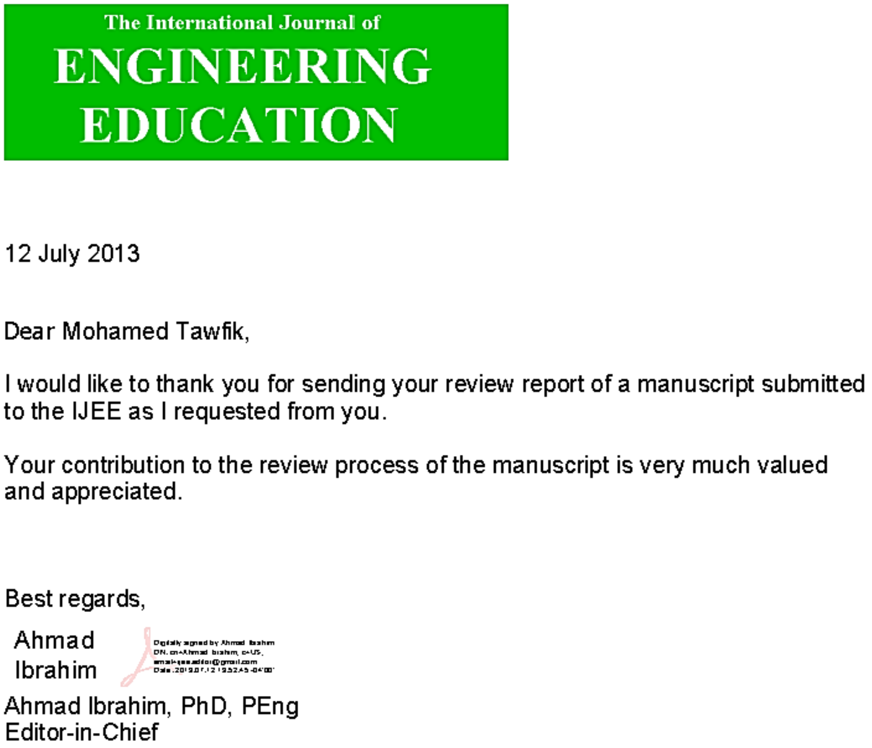
DR. D. MANUEL-ALONSO CASTRO GIL, Presidente de IEEE SECCIÓN ESPAÑA,

HACE CONSTAR: que **D. MOHAMED TAWFIK**, estudiante de la Universidad Nacional de Educación a Distancia y Secretario/Tesorero de la Rama de Estudiantes del IEEE en la UNED, **ha colaborado como coordinador de actividades de diseminación web dentro de la Directiva de IEEE SECCIÓN ESPAÑA**, en el periodo desde: 01/10/2010 hasta: 01/01/2012.

Y para que conste y surta los efectos oportunos, firmo la presente en Madrid, a tres de enero de 2012.

FDO. : MANUEL-ALONSO CASTRO GIL

Presidente de IEEE SECCIÓN ESPAÑA



The International Journal of
ENGINEERING
EDUCATION

12 July 2013

Dear Mohamed Tawfik,

I would like to thank you for sending your review report of a manuscript submitted to the IJEE as I requested from you.

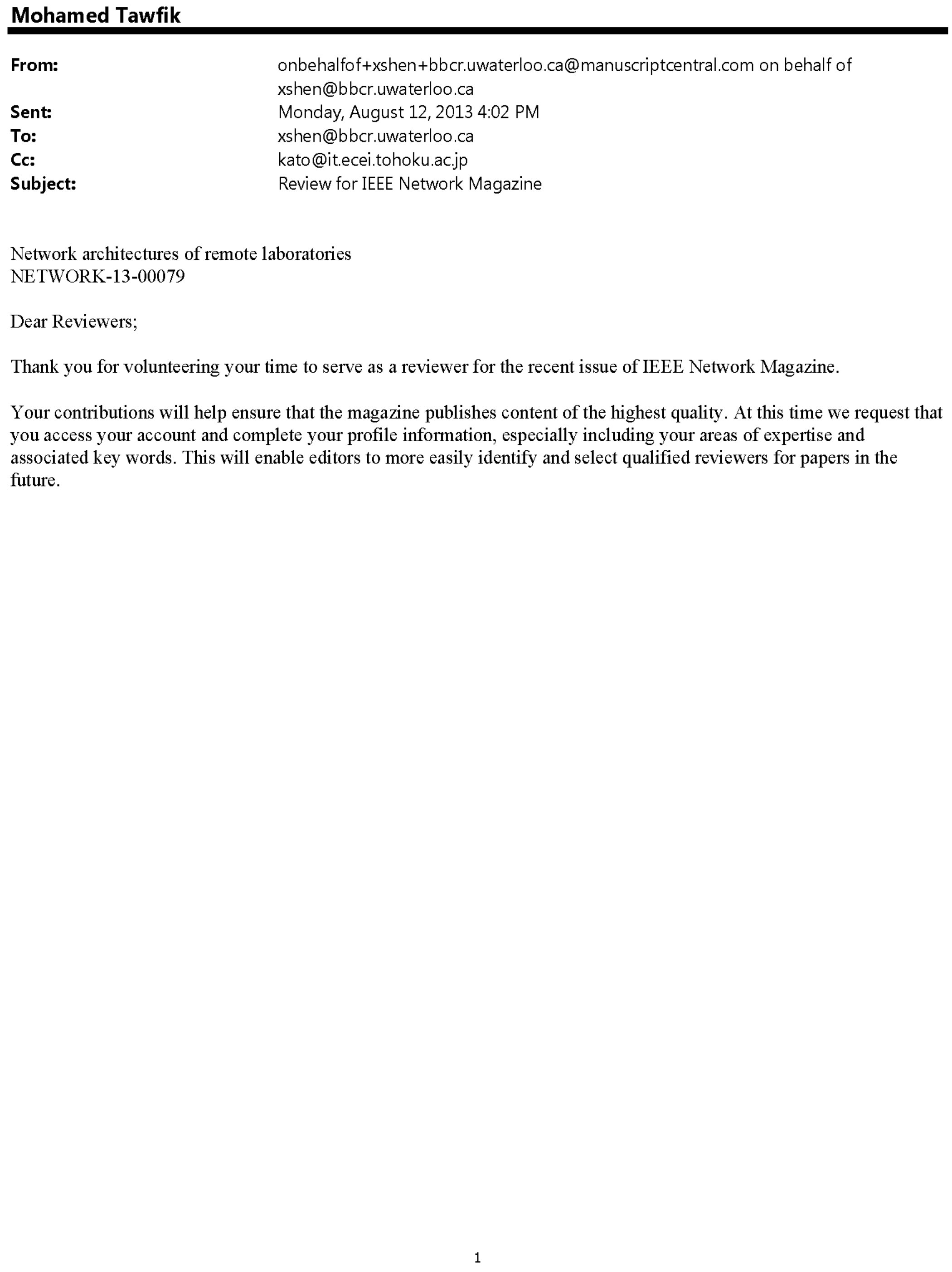
Your contribution to the review process of the manuscript is very much valued and appreciated.

Best regards,

Ahmad
Ibrahim

Digitally signed by Ahmad Ibrahim
DN: cn=Ahmad Ibrahim, c=US,
email=ijee.editor@gmail.com
Date: 2013.07.12 19:52:45 -04'00'

Ahmad Ibrahim, PhD, PEng
Editor-in-Chief



Mohamed Tawfik

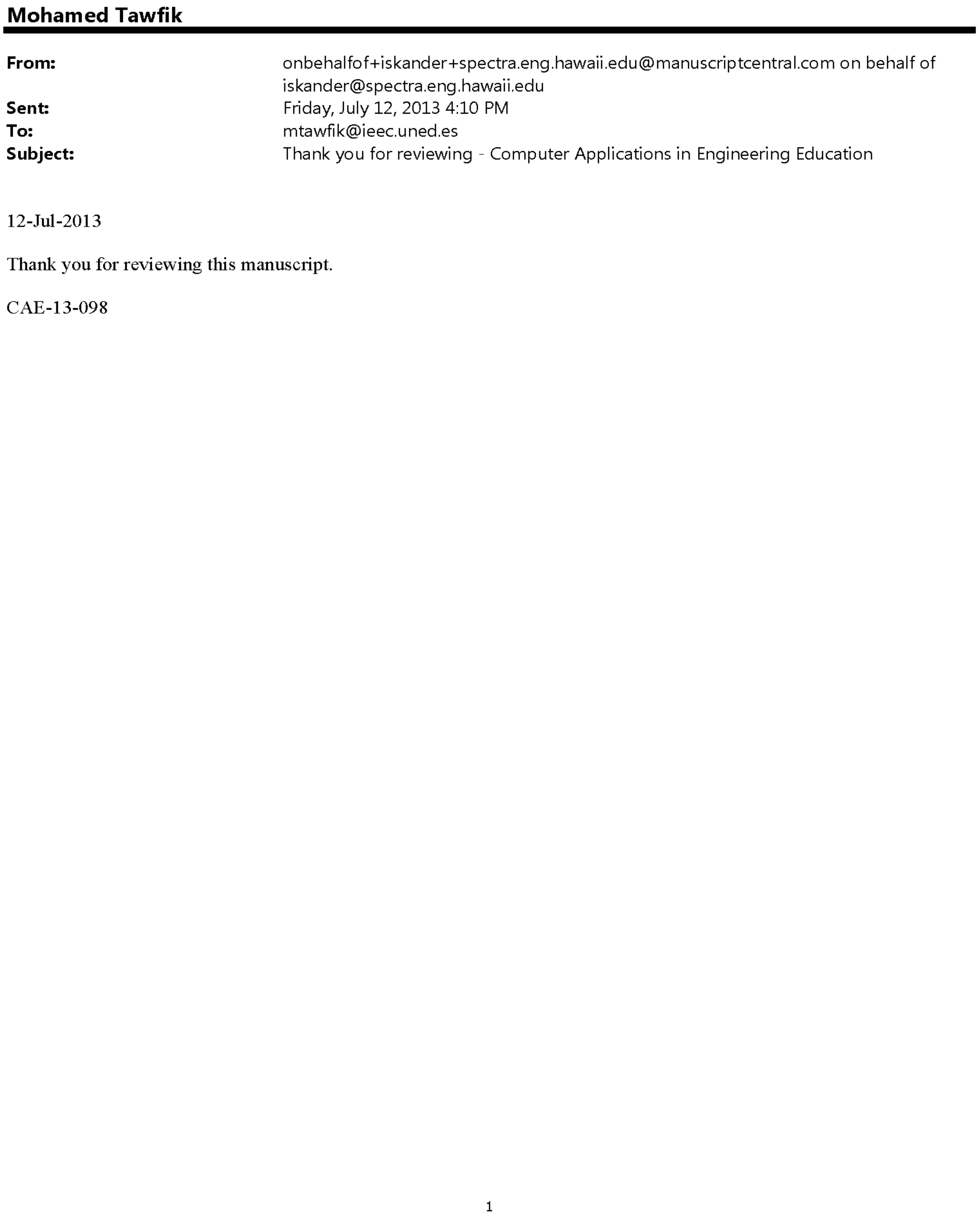
From: onbehalfof+xshen+bbcr.uwaterloo.ca@manuscriptcentral.com on behalf of xshen@bbcr.uwaterloo.ca
Sent: Monday, August 12, 2013 4:02 PM
To: xshen@bbcr.uwaterloo.ca
Cc: kato@it.ecei.tohoku.ac.jp
Subject: Review for IEEE Network Magazine

Network architectures of remote laboratories
NETWORK-13-00079

Dear Reviewers;

Thank you for volunteering your time to serve as a reviewer for the recent issue of IEEE Network Magazine.

Your contributions will help ensure that the magazine publishes content of the highest quality. At this time we request that you access your account and complete your profile information, especially including your areas of expertise and associated key words. This will enable editors to more easily identify and select qualified reviewers for papers in the future.



Mohamed Tawfik

From:	onbehalfof+iskander+spectra.eng.hawaii.edu@manuscriptcentral.com on behalf of iskander@spectra.eng.hawaii.edu
Sent:	Friday, July 12, 2013 4:10 PM
To:	mtawfik@ieec.uned.es
Subject:	Thank you for reviewing - Computer Applications in Engineering Education

12-Jul-2013

Thank you for reviewing this manuscript.

CAE-13-098

This page intentionally left blank.

Appendix F

F. Additional References

B. M. Wilamowski and J. D. Irwin, *The Industrial Electronics Handbook Series* (Fundamentals of Industrial Electronics, Power Electronics and Motor Drives, Control and Mechatronics, Industrial Communication Systems, and Intelligent Systems), Second ed.: Taylor & Francis Group, LLC (CRC), 2011

C. R. Robertson, *Fundamental Electrical and Electronic Principles*, Third ed.: Elsevier Ltd (Newnes), 2008.

D. M. Solis, *Illustrated C#* Apress, 2010.

E. R. C. Dorf, *The Electrical Engineering Handbook* (Electronics, Power Electronics, Optoelectronics, Microwaves, Electromagnetics, and Radar), Third ed.: Taylor & Francis Group, LLC (CRC), 2006.

G. Rizzoni, *Fundmentals of Electrical Engineering*: McGraw-Hill Companies, Inc., 2009.

J. Bird, *Higher Engineering Mathematics*, Fifth ed.: Elsevier Ltd. (Newnes), 2006.

J. Kring, *LabVIEW for Everyone: Graphical Programming Made Easy and Fun*, Third ed.: Prentice Hall, 2006.

J. Fawcett, L. R. E. Quin, and D. Ayers, *Beginning XML*, Fifth ed.: John Wiley & Sons, Inc., 2012.

J. Duckett, *Beginning HTML, XHTML, CSS, and JavaScript*: Wiley Publishing, Inc., 2010.

J. Bird, *Electrical and Electronic Principles and Technology*, Fourth ed.: Elsevier Ltd. (Newnes), 2010.

LabVIEWTM Intermediate II, Connectivity Course Manual, Course Software Version 8.2, February 2007 Edition, Part Number 323758C-01.

Online video tutorials & training | lynda.com (www.lynda.com), 2013

M. A. Laughton and D. J. Warne, *Electrical Engineer's Reference Book*, Sixteenth ed.: Elsevier Science (Newnes), 2003.

N. S. Nise, *Control System Engineering* Sixth ed.: John Wiley & Sons, Inc, 2010

P. Scherz, *Practical Electronics for Inventors*: McGraw-Hill, 2000.

P. Zhang, *Advanced Industrial Control Technology*: Elsevier Inc. (William Andrew), 2010.

P. Wilton and J. McPeak, *Beginning JavaScript*, Fourth ed.: Wiley Publishing, Inc., 2010.

P. MacIntyre, B. Danchilla, and M. Gogala, *Pro PHP Programming* Apress, 2011.

R. Boylestad and L. Nashelsky, *Electronic Devices and Circuit Theory*, Ninth ed.: Pearson Prentice Hall, 2006.

S. Holzner, *Ajax: A Beginner's Guide*: The McGraw-Hill Companies, 2009.

S. Casteleyn, F. Daniel, P. Dolog, and M. Matera, *Engineering Web Applications*. Springer Dordrecht Heidelberg London New York: Springer, 2009.

T. L. Floyd, *Digital fundamentals*, Ninth ed.: Pearson Education Ltd., 2006.

T. L. Floyd, *Electronic Devices: Electron Flow Version*, Ninth ed.: Pearson Education Ltd, 2012.

T. L. Floyd, *Principles of Electric Circuits: Conventional Current Version*, Eight ed.: Pearson Education Ltd., 2007.

Vaswani, *Zend Framework: A Beginner's Guide*, The McGraw-Hill Companies, 2010.

YouTube (www.youtube.com/), 2013

W. J. Gilmore, *Beginning PHP and MySQL from Novice to Professional*, Fourth ed.: Apress, 2010.

Wikipedia (www.wikipedia.org/), 2013.

References

[1] A. P. Carnevale, N. Smith, and M. Melton, "STEM: Science Technology Engineering Mathematics," The Georgetown University Center on Education and the Workforce, Washington, DC 20007, October 20, 2011.

[2] J. E. Froyd, P. C. Wankat, and K. A. Smith, "Five Major Shifts in 100 Years of Engineering Education," *Proceedings of the IEEE*, vol. 100, Special Centennial Issue, pp. 17, May 13th, 2012.

[3] J. Bourne, D. Harris, and F. Mayadas, "Online Engineering Education: Learning Anywhere, Anytime," *Journal of Engineering Education,* vol. 94, no. 1, pp. 16, January, 2005.

[4] I. E. Allen and J. Seaman, "The ninth annual survey of Solan-C: Going the Distance: Online Education in the United States, 2011," The Sloan Consortium, Newburyport 2011.

[5] L. D. Feisel and A. J. Rosa, "The Role of the Laboratory in Undergraduate Engineering Education," *Journal of Engineering Education,* vol. 94, no. 1, pp. 10, January, 2005.

[6] E. Hanen, "The role of interactive video technology in higher education: case study and a proposed framework," *Educational Technology* vol. 30, no. 9, pp. 9, September 1990.

[7] P. C. Wankat, "Analysis of the First Ten Years of the Journal of Engineering Education," *Journal of Engineering Education,* vol. 93, no. 1, pp. 9, January, 2004.

[8] M. F. Aburdene, E. J. Mastascusa, and R. Massengale, "A proposal for a remotely shared control systems laboratory," presented at the Frontiers in Education Conference, 1991. Twenty-First Annual Conference. 'Engineering Education in a New World Order.' Proceedings., West Lafayette, IN, 1991, pp. 589 - 592

[9] B. Aktan, C. A. Bohus, L. A. Crowl, and M. H. Shor, "Distance Learning Applied to Control Engineering Laboratories," *IEEE Transactions on Education,* vol. 39, no. 3, pp. 7, August, 1996.

[10] B. Carisa, A. Burain, S. M. H., and C. Lawrence, "Running Control Engineering Experiments Over the Internet," Department of Computer Science, Oregon State University Corvallis, 1995.

[11] J. E. Corter, J. V. Nickerson, S. K. Esche, C. Chassapis, S. Im, and J. Ma, "Constructing reality: A study of remote, hands-on, and simulated laboratories," *ACM Transactions on Computer-Human Interaction (TOCHI),* vol. 14, no. 2, pp. 7, 2007.

[12] E. D. Lindsay and M. C. Good, “Effects of laboratory access modes upon learning outcomes,” *Education, IEEE Transactions on,* vol. 48, no. 4, pp. 13, November, 2005.

[13] J. E. Corter, S. K. Esche, C. Chassapis, J. Ma, and J. V. Nickerson, “Process and learning outcomes from remotely-operated, simulated, and hands-on student laboratories,” *Computers & Education,* vol. 57, no. 3, pp. 14, November, 2011.

[14] D. Lowe, G. Bharathy, B. Stumpers, and H. Yeung, "Laboratory Lesson Plans: Opportunities Created by Remote Laboratories," presented at the Remote Engineering and Virtual Instrumentation (REV), 9th International Conference on, Bilbao, 2012, p. 6.

[15] A. Mujkanovic, D. Lowe, K. Willey, and C. Guet, "Unsupervised Learning Algorithm for Adaptive Group Formation: Collaborative Learning Support in Remotely Accessible Laboratories," presented at the International Conference on Information Society (i-Society), London, 2012, p. 8.

[16] S. Farah, A. Benachenhou, G. Neveux, and D. Barataud, “Design of a Flexible Hardware Interface for Multiple Remote Electronic Practical Experiments of Virtual Laboratory,” *International Journal of Online Engineering (iJOE),* vol. 8, no. 2, pp. 6, March, 2012.

[17] D. A. H. Samuelsen and O. H. Graven, “Low Cost Implementation of Remote Lab with Large Number of Configurations for a BJT Amplifier,” *International Journal of Online Engineering (iJOE),* vol. 7, no. 1, pp. 6, September, 2011.

[18] N. Sousa, G. R. Alves, and M. G. Gericota, “An Integrated Reusable Remote Laboratory to Complement Electronics Teaching,” *Learning Technologies., IEEE Transaction on,* vol. 3, no. 3, pp. 7, July-September, 2010.

[19] Z. Nedic, J. Machotka, and A. Nafalski, "Remote Laboratory NetLab for Effective Interaction with Real Equipment over the Internet," presented at the Human System Interactions, Conference on, Krakow, 2008, p. 6.

[20] D.V.Pop, M.E.Auer, and D.G.Zutin, "New Solution for Remote Design and Test of Digital Systems with the Altera MAX CPLD Laboratory within the iLab Shared Architecture," presented at the 10th International Conference on Remote Engineering and Virtual Instrumentation (REV), Brasov, Romania, 2011, p. 6.

[21] W. A. Pleskacz, V. Stopjaková, T. Borejko, A. Jutman, and A. Wałkanis, “DefSim: A Remote Laboratory for Studying Physical Defects in CMOS Digital Circuits,” *IEEE Transactions on Industrial Electronics,* vol. 55, no. 6, pp. 11, June, 2008.

[22] A. Mohtar, Z. Nedic, and J. Machotka, "A Remote Laboratory for Microelectronics Fabrication," presented at the 38th ASEE/IEEE Frontiers in Education (FIE) Conference Saratoga Springs, NY, 2008, p. 6.

[23] A. P. J. Chandra and C. R. Venugopal, “Novel Design Solutions for Remote Access, Acquire and Control of Laboratory Experiments on DC Machines,” *IEEE Transactions on Instrumentation and Measurements,* vol. 61, no. 2, pp. 9, February, 2012.

[24] E. Irmak, R. Bayindir, I. Colak, and M. Soysal, “A Remote Laboratory Experiment for 4-Quadrant Control of a DC Motor,” *Computer Applications in Engineering Education,* vol. 19, no. 4, pp. 14, December, 2011.

[25] A. Tekin, F. Ata, and M. Gökbulut, “Remote Control Laboratory for DSP-Controlled Induction Motor Drives,” *Computer Applications in Engineering Education,* vol. 20, no. 4, pp. 11, December, 2012.

[26] P. N. Borza, P. A. Cotfas, D. T. Cotfas, and M. C. Carp, "PV Cells Test Bench System with Remote Access Through Internet," presented at the Optimization of Electrical and Electronic Equipment (OPTIM), 2012 13th International Conference on, Brasov, Romania 2012, p. 6.

[27] I. Khan, D. Muthusamy, W. Ahmad, B. Sällberg, K. Nilsson, J. Zackrisson, *et al.*, “Performing Active Noise Control and Acoustic Experiments Remotely,” *International Journal of Online Engineering (iJOE),* vol. 8, no. 4, pp. 10, December, 2012.

[28] J. C. A.P and C. R. Venugopal, “Web based Access and Control Environment for Experiment on Electronic Circuits,” *International Conference on Computer & Communication Technology (ICCCT),* pp. 5, September 15-17, 2011.

[29] H. Kyomugisha, T. Kigezi, C. Mwikirize, R. Akol, D. Orishaba, and M. Kyesswa, “Remote Direct Sequence Spread Spectrum Communications Lab Utilising the Emona DATEx,” *International Journal of Online Engineering (iJOE),* vol. 8, no. 4, pp. 5, December, 2012.

[30] O. O. Aboluwarin, K. P. Ayodele, L. O. K. P.E., and B. I. Ishola, "Remote Realistic Interface Experimentation Using the Emona DATEx Board," presented at the 119th ASEE Annual Conference & Exposition San Antonio, USA, 2012, p. 17.

[31] M. T. Restivo, I. Member, J. Mendes, A. M. Lopes, C. M. Silva, and F. Chouzal, “A Remote Laboratory in Engineering Measurement,” *IEEE Transactions on Industrial Electronics,* vol. 56, no. 12, pp. 8, December 12, 2009.

[32] AshishMani, R. S. S. Prasanth, and C. Patvardhan, “Design ofWeb-Based Experiments on Acceleration and Speed Transducers,” *Education Research International,* vol. 2011, ID:426062, pp. 7, 2011.

[33] J. J. Fuertes, S. ı. Alonso, A. Moran, M. A. Prada, S. ı. Garcia, and C. d. Canto, "Virtual and Remote Laboratory of a DC Motor," presented at the Proceedings of the 9th IFAC Symposium Advances in Control Education, Nizhny Novgorod, Russia, 2012, p. 6.

[34] C. Ferrater-Simon, L. Molas-Balada, O. Gomis-Bellmunt, N. Lorenzo-Martinez, O. Bayo-Puxan, and R. Villafafila-Robles, “A Remote Laboratory Platform for Electrical Drive Control Using Programmable Logic Controllers,” *Education, IEEE Transactions on,* vol. 52, no. 3, pp. 11, August, 2009.

[35] M. Domínguez, J. J. Fuertes, M. A. Prada, S. Alonso, and A. Morán, “Remote Laboratory of a Quadruple Tank Process for Learning in Control Engineering Using Different Industrial Controllers,” *Computer Applications in Engineering Education,* Early view, pp. 12, June.

[36] M. Domíez, J. J. Fuertes, P. Reguera, A. Morán, S. Alonso, and M. A. Prada, "Remote Laboratory for Learning of AC Drive Control," presented at the Proceedings of the 18th World Congress The International Federation of Automatic Control, Milan, 2011, p. 6.

[37] F. Y. Limpraptono, A. A. P. Ratna, and H. Sudibyo, "Remote Laboratories Multiuser Based on Embedded Web server," presented at the Remote Engineering and Virtual Instrumentation (REV), 9th International Conference on Bilbao, 2012, p. 7.

[38] D. Chaos, J. Chacon, J. A. Lopez-Orozco, and S. Dormido, “Virtual and Remote Robotic Laboratory Using EJS, MATLAB and LabVIEW,” *Sensors,* vol. 13, no. 2, pp. 18, February 21, 2013.

[39] R. Purohit, S. Sahu, and S. Kant, “Intelligent Cardiac Monitoring System Using Virtual Instrumentation Techniques,” *International Journal of Online Engineering (iJOE),* vol. 7, no. 3, pp. 5, June, 2011.

[40] I. Gustavsson, K. Nilsson, J. Zackrisson, J. Garcia-Zubia, U. Hernandez-Jayo, A. Nafalski, *et al.*, “On Objectives of Instructional Laboratories, Individual Assessment, and Use of Collaborative Remote Laboratories,” *IEEE Transactions on Learning Technologies* vol. 2, no. 4, pp. 12, Oct.-Dec., 2009.

[41] M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, J. Peire, *et al.*, “Virtual Instrument Systems in Reality (VISIR) for Remote Wiring and Measurement of Electronic Circuits on Breadboard,” *Learning Technologies, IEEE Transactions on,* vol. 6, no. 1, pp. 13, March 12 (First Quarter) 2013.

[42] (2013, May). *Special Interest Group of VISIR (SIG VISIR),* Available: http://www.online-engineering.org/index.php?option=com_content&view=category&layout=blog&id=42&Itemid=74

[43] U. H. Jayo, "Metodología de Control Independiente de Instrumentos y Experimentos para su Despliegue en Laboratorios Remotos" Doctoral Thesis, Facultad de Ingenieria, Universidad de Deusto, 2012.

[44] M. Tawfik, E. Sancristobal, S. Martin, C. Gil, A. Pesquera, P. Losada, *et al.*, "VISIR Deployment in Undergraduate Engineering Practices," presented at the Frontiers in Education Conference (FIE) - 1st Global Online Laboratory Consortium (GOLC) Workshop- Rapid City, South Dakota, USA, 2011, p. 7.

[45] UNED-COMA. (2013, May). *Bases de Circuitos y Electrónica Práctica*. Available: https://unedcoma.es/course/bases-de-circuitos-y-electronica-practica/

[46] MITx. (2013, May). *Circuits & Electronics 6.002x* Available: https://6002x.mitx.mit.edu/

[47] (2013, May). *RIPLECS Project - Remote-labs access in Internet-based Performance-centred Learning Environment for Curriculum Support*. Available: http://riplecs.dipseil.net/

[48] M. Tawfik, E. Sancristobal, S. Martin, R. Gil, A. Pesquera, M. J. Albert, *et al.*, "Labor-Oriented Online Master Degree Program," presented at the International Conference on Remote Engineering and Virtual Instrumentation (REV), Bilbao, Spain, 2012, pp. 108-114.

[49] M. Tawfik, E. Sancristobal, S. Martin, R. Gil, A. Pesquera, M. J. Albert, *et al.*, “European Online Master Degree Program for Addressing Labor Market Demands,” *International Journal of Online Engineering (iJOE),* vol. 8, no. 3, pp. 7, November, 2012.

[50] M. Tawfik, E. S. Cristobal, A. Pesquera, R. Gil, S. Martin, G. Diaz, *et al.*, "Shareable Educational Architectures for Remote Laboratories," presented at the Technologies Applied to Electronics Teaching (TAEE), 2012, Vigo, Spain, 2012, pp. 122-127.

[51] M. Niederstätter and C. Maier, "A Semantic Portal for Publication and Exchange of Educational Online Laboratories " in *Internet Accessible Remote Laboratories: Scalable E-Learning Tools for Engineering and Science Disciplines*, ed: IGI Global 2012, p. 18.

[52] V. Mateos, A. Gallardo, T. Richter, L. Bellido, P. Debicki, and V. A. Villagrá, “LiLa Booking System: Architecture and Conceptual Model of a Rig Booking System for On-Line Laboratories,” *International Journal of Online Engineering (iJOE),* vol. 7, no. 4, pp. 10, 2011.

[53] E. Sancristobal, "Metodología, Estructura y Desarrollo de Interfaces Intermedias para la Conexión de Laboratorios Remotos y Virtuales a Plataformas Educativas" Doctoral Thesis, Electrical & Computer Engineering Department (DIEEC), Spanish University for Distance Education (UNED), Madrid, 2010.

[54] E. Sancristobal, M. Castro, J. Harward, P. Baley, K. DeLong, and J. Hardison, "Integration view of Web Labs and Learning Management Systems," in *IEEE Global Eng. Educ. Con. (EDUCON)* 2010, pp. 1409-1417.

[55] E. Bogdanov, S. E. Helou, D. Gillet, C. Salzmann, and S. Sire, "Graaasp: a Web 2.0 Research Platform for Contextual Recommendation with Aggregated Data," presented at the CHI 2010: Work-in-Progress (Spotlight on Posters Days 1 & 2), Atlanta, GA, USA, 2010.

[56] B. Evgeny, S. Christophe, and D. Gillet, "Widget-Based Approach for Remote Control Labs," presented at the 9th IFAC Symposium on Advances in Control Education, Resort Automobilist, Russia, 2012.

[57] V. J. Harward, J. A. del Alamo, S. R. Lerman, P. H. Bailey, J. Carpenter, K. DeLong, *et al.*, “The iLab Shared Architecture: A Web Services Infrastructure to Build Communities of Internet Accessible Laboratories,” *Proceedings of the IEEE,* vol. 96, no. 6, pp. 20, June, 2008.

[58] M. Tawfik, D. Lowe, S. Murray, M. d. l. Villefromoy, M. Diponio, E. Sancristobal, *et al.*, "Grid Remote Laboratory Management System: Sahara Reaches Europe," presented at the 10th International Conference on Remote Engineering and Virtual Instrumentation (REV), Sydney, 2013, p. 9.

[59] P. Orduna, "Transitive and Scalable Federation Model for Remote Laboratories" Doctoral Thesis, University of Deusto, Bilbao, 2013.

[60] (2013, May). *MIT iCampus: iLabs*. Available: http://ilab.mit.edu/wiki

[61] (2013, June). *Labshare - Home*. Available: http://www.labshare.edu.au/

[62] D. Lowe, T. Machet, and T. Kostulski, "UTS Remote Labs, Labshare, and the Sahara Architecture," in *Using Remote Labs in Education: Two Little Ducks in Remote Experimentation*, J. G. Zubía and G. R. Alves, Eds., ed University of Deustoo, 2011, p. 22.

[63] D. Lowe, S. Murray, E. Lindsay, and L. Dikai, “Evolving Remote Laboratory Architectures to Leverage Emerging Internet Technologies,” *Learning Technologies, IEEE Transactions on,* vol. 2, no. 4, pp. 6, 2009.

[64] (2013, June). *Global Online Laboratory Consortium (GOLC)* Available: http://online-lab.org/

[65] H. Yeung, D. Lowe, and S. Murray, “Interoperability of Remote Laboratories Systems,” *International Journal of Online Engineering (iJOE),* vol. 6, pp. 10, 2010.

[66] M. Tawfik, E. Sancristobal, S. Martin, G. Diaz, J. Peire, and M. Castro, “Expanding the Boundaries of the Classroom: Implementation of Remote Laboratories for Industrial Electronics Disciplines,” *Industrial Electronics Magazine, IEEE,* vol. 7, no. 1, pp. 9, March 19, 2013.

[67] E. Sancristobal, P. Orduña, M. Tawfik, F. García, O. Dziabenko, D. López-de-Ipiña, *et al.*, "Widget and smart devices. A different aproach for online learning scenarios," presented at the IEEE Global Engineering Education Conference (EDUCON), Berlin, Germany, 2013, pp. 808 - 812.

[68] Y. Yuhong, L. Yong, D. Xinge, H. S. Hassane, and A. Ghorbani, “Putting labs online with Web services,” *IT Professional,* vol. 8, no. 2, pp. 8, March-Abril, 2006.

[69] C. D. Capua, A. Liccardo, and R. Morello, "On the Web Service-Based Remote Didactical Laboratory: Further Developments and Improvements," presented at the Instrumentation and Measurement Technology Conference (IMTC), Proceedings of the IEEE (Volume:3) Ottawa, Ont., 2005, p. 5.

[70] A. Baccigalupi, C. D. Capua, and A. Liccardo, "Overview on Development of Remote Teaching Laboratories: from LabVIEW to Web Services," presented at the Instrumentation and Measurement Technology Conference (IMTC), Sorrento, Italy, 2006, p. 6.

[71] M. Ngolo, L. B. Palma, F. Coito, L. Gomes, and A. Costa, "Architecture for remote laboratories based on REST web services," presented at the E-Learning in Industrial Electronics. ICELIE '09. 3rd IEEE International Conference on Porto 2009, p. 6.

[72] S. Dutta, S. Prakash, D. Estrada, and E. Pop, “A Web Service and Interface for Remote Electronic Device Characterization,” *Education, IEEE Transactions on,* vol. 54, no. 4, pp. 6, November, 2011.

[73] C. Salzmann and D. Gillet, "Smart device paradigm, Standardization for online labs," presented at the Global Engineering Education Conference (EDUCON), IEEE, Berlin, 2013, p. 5.

[74] V. Cheruku, "Integrating Physical Laboratories into a Cloud Environment," Master Thesis, Computer Science, North Carolina State University, Raleigh, North Carolina, 2013.

[75] R. I. Dinita, G. Wilson, A. Winckles, M. Cirstea, and A. Jones, "A Cloud-based Virtual Computing Laboratory for Teaching Computer Networks," presented at the Optimization of Electrical and Electronic Equipment (OPTIM), 13th International Conference on, Brasov, 2013, p. 5.

[76] C. R. S. d. Oliveira and I. N. d. Oliveira, "Uma proposta para a disponibilidade de Laboratórios de Física como serviços da Computação em Nuvem," presented at the 9th International Information and Telecommunication Technologies Symposium (I2TS'2010) Rio de Janeiro, Brazil, 2010, p. 4.

[77] J. Rugelj, M. Ciglarič, A. Krevl, M. Pančur, and A. Brodnik, "Constructivist Learning Environment in a Cloud," in *Workshop on Learning Technology for Education in Cloud (LTEC'12)*. vol. 173, L. Uden, E. S. Corchado Rodríguez, J. F. De Paz Santana, and F. De la Prieta, Eds., ed: Springer Berlin Heidelberg, 2012, pp. 193-204.

[78] Y. Luo and X. Q. Zhang, "Cloud-Based Platform Building Research of Teaching Resources," *Applied Mechanics and Materials*, vol. 347-350, pp. 4, August, 2013.

[79] S. Seiler, "Laboratory as a Service A Holistic Framework for Remote and Virtual Labs" Doctoral Thesis, Faculty of Mechanical Engineering-Department of Mechatronics, Tallinn University of Technology, 2012.

[80] M. Castro, "One Step Ahead in the Future of Labs: Widgets, Ubiquity and Mobility," presented at the International Conference on Remote Engineering and Virtual Instrumentation (REV)-Keynote, Bilbao, Spain, 2012, pp. 1-10.

[81] S. Bitzer, S. Ramroth, and M. Schumann, "Mashups as an Architecture for Knowledge Management Systems," presented at the HICSS '09. 42nd Hawaii International Conference on System Sciences, Big Island, HI 2009, p. 10.

[82] D. Dagger, A. O'Connor, S. Lawless, E. Walsh, and V. P. Wade, "Service-Oriented E-Learning Platforms: From Monolithic Systems to Flexible Services," *Internet Computing, IEEE*, vol. 11, no. 3, pp. 8, May-June, 2007.

[83] M. A. C. Gonzalez, F. J. G. Penalvo, M. J. C. Guerrero, and M. A. Forment, "Adapting LMS Architecture to the SOA: An Architectural Approach," in *Internet and Web Applications and Services, 2009. ICIW '09. Fourth International Conference on*, Venice/Mestre, 2009, p. 6.

[84] M. J. C. Guerrero, M. A. Forment, M. A. C. Gonzalez, and F. J. G. Penalvo, "SOA Initiatives for eLearning: A Moodle Case," in *Advanced Information Networking and Applications Workshops, 2009. WAINA '09. International Conference on*, Bradford, 2009, p. 6.

[85] "Native, Web or Hybrid Mobile-App Development," *IBM Software - Thought Leadership White Paper*, vol. IBM Corporation-Software Group, April, 2012.

[86] L. E. Moser and P. M. Melliar-Smith, "Service-Oriented Architecture and Web Services," in *Wiley Encyclopedia of Computer Science and Engineering*, B. W. Wah, Ed., ed: John Wiley & Sons, Inc, 2008, p. 8.

[87] C. Pautasso, O. Zimmermann, and F. Leymann, "RESTful Web Services vs. “Big” Web Services: Making the Right Architectural Decision," presented at the WWW '08 Proceedings of the 17th international conference on World Wide Web Beijing, China, 2008, p. 10.

[88] M. Tawfik, E. Sancristobal, S. Martin, R. Gil, G. Diaz, J. Peire, *et al.*, "On the Design of Remote Laboratories," presented at the IEEE Global Engineering Education Conference (EDUCON), Marrakesh, Morocco, 2012, pp. 311-316.

[89] A. Chenakin, “Control Interfaces for RF and Microwave Frequency Synthesizers,” *High Frequency Electronics,* vol. CONTROL INTERFACES, pp. 7, February, 2010.

[90] N. I. Corporation, “Designing Next Generation Test Systems: An In-Depth Developers Guide,” no. www.ni.com/white-paper/3238/en, pp. 39, Abril, 2013.

[91] W. Su and F. Wang, "Design and Development of Rolling Bearing Vibration Signal Analysis System," presented at the International Conference on Electronic & Mechanical Engineering and Information Technology, Harbin, Heilongjiang, China 2011.

[92] A. Rojko, D. Hercog, and K. Jezernik, “Power Engineering and Motion Control Web Laboratory: Design, Implementation, and Evaluation of Mechatronics Course,” *IEEE Transactions on Industrial Electronics,* vol. 57, no. 10, pp. 12, October, 2010.

[93] D. Hercog, B. Gergic, S. Uran, and K. Jezernik, “A DSP-Based Remote Control Laboratory,” *IEEE Transaction on Industrial Electronics,* vol. 54, no. 6, pp. 12, December, 2007.

[94] M. Tawfik, E. Sancristobal, S. Martin, G. Diaz, and M. Castro, "State-of-the-art Remote Laboratories for Industrial Electronics Applications," presented at the Technologies Applied to Electronics Teaching (TAEE), 2012, pp. 359-364.

[95] L. Costas-Perez, D. Lago, J. Farina, and J. J. Rodriguez-Andina, “Optimization of an Industrial Sensor and Data Acquisition Laboratory Through Time Sharing and Remote Access,” *Industrial Electronics, IEEE Transactions on* vol. 55, no. 6, pp. 8, June, 2008.

[96] C. A. Jara, F. A. Candelas, P. Gil, F. Torres, F. Esquembre, and S. Dormido, “EJS+EjsRL: An interactive tool for industrial robots simulation, Computer Vision and remote operation,” *Robotics and Autonomous Systems,* vol. 59, no. 6, pp. 13, June, 2011.

[97] H. Vargas, J. Sánchez-Moreno, S. Dormido, C. Salzmann, D. Gillet, and F. Esquembre, “Web-Enabled Remote Scientific Environments ” *Computing in Science & Engineering,* vol. 11, no. 3, pp. 11, May-June 2009.

[98] G. Farias, R. D. Keyser, S. Dormido, and F. Esquembre, “Developing Networked Control Labs: A Matlab and Easy Java Simulations Approach,” *IEEE Transactions on Industrial Electronics,* vol. 57, no. 10, pp. 10, October, 2010.

[99] S. Müller and H. Waller, "Efficient Integration Of Real-Time Hardware and Web Based Services Into MATLAB," presented at the Proceedings of 11th European Simulation Symposium, Erlangen, Germany, 1999, p. 5.

[100] P. Bisták and P. Folvar, “Remote Laboratory Java Server Based on JACOB Project,” *International Journal of Online Engineering (iJOE),* vol. 7, no. 1, pp. 4, February, 2011.

[101] M. Ngolo, L. B. Palma, F. Coito, L. Gomes, and A. Costa, "Architecture for remote laboratories based on REST web services," in *E-Learning in Industrial Electronics, 2009. ICELIE '09. 3rd IEEE International Conference on*, Porto, 2009, p. 6.

[102] (2013, September). *N. I. Corporation, NI USB-6008/6009 User Guide and Specifications*. Available: www.ni.com/pdf/manuals/371303m.pdf

[103] M. Tawfik, "VISIR Installation and Start-Up Guide," Master Thesis, Electrical & Computer Engineering Department (DIEEC), Spanish University for Distance Education (UNED), Madrid, 2011.