Supporting Local Access to Collections of Distributed Remote Laboratories

Michael Diponio[a], David Lowe [b], and Michel de la Villefromoy[a]

[a]*Centre for Real-Time Information Network, Faculty of Engineering and IT*
University of Technology, Sydney
[b]*Faculty of Engineering and Information Technologies*
The University of Sydney

Corresponding Author Email: david.lowe@uts.edu.au

BACKGROUND
Remote laboratories represent a novel resource in supporting improved access to a diverse range of educational laboratory experiences. An important benefit arising from remote laboratories is the ability to share laboratories across institutional boundaries. The extent of sharing (and hence the potential benefits) is however strongly affected by the ease with which the lab owner can coordinate access, and the ease with which users can gain access. Federated access technologies such as Shibboleth (and its implementation within the Australian Access Federation) provide a partial solution, but typically still require users to visit the lab owner's site in order to gain access. The resulting challenges in lab branding and diverse interfaces can still result in a hurdle to widespread sharing.

PURPOSE
This paper considers the question of whether it is feasible to support a richer form of federated access to remotely accessed teaching laboratories, whereby users can transparently access, through their own local system, a range of apparatus drawn from diverse external sources. Further, if such an approach is feasible, then what are the system architectural implications and can such support be seamlessly integrated into an existing remote laboratory system?

DESIGN/METHOD
Potential architectures that can support local access to distributed remote laboratories are identified – including centralised, distributed and peer-to-peer architectures. Using a lens of contextual (existing and legacy laboratories, inclusiveness), operational (ownership, access negotiation, user perception) and technical (performance, scalability, security) factors these architectures are critiqued. An architecture has subsequently been chosen and used as the basis for a prototype system based on the existing Sahara remote laboratory system. The resulting prototype was evaluated in terms of its ability to provide the desired functionality.

RESULTS
The results clearly demonstrate the benefits of extending an existing remote laboratory system so that distributed remote laboratory resources can be aggregated and accessed from a single location. Evaluations of the prototype have shown no major technical or performance problems as well as the potential for significant improvements in user access to these resources.

CONCLUSIONS
This paper has successfully shown that it is feasible to extend an existing remote laboratory system to support local management of access to an aggregated collection of remote labs drawn from diverse locations. This has the potential to simplify the process of access and to therefore encourage more widespread resource sharing. Future work is still required to fully test the implementation and integrate it into the current Sahara production release, but this is anticipated to be a relatively straightforward process. More significantly, further work will also more broadly consider how sharing of remote labs is encouraged, funded, and especially used as a vehicle for pedagogic innovation and sharing of best practice.

KEYWORDS
Remote Laboratories; Sharing; Federation.

Introduction

Remotely accessible laboratories (or “remote labs”) represent a novel resource in supporting improved access to a diverse range of educational laboratory experiences. A remote lab is based on instrumenting a physical laboratory apparatus so that it can be monitored and controlled via the internet, with the result that users need not be physically present with the apparatus. An example of a typical remote laboratory interface is provided in Figure 1.

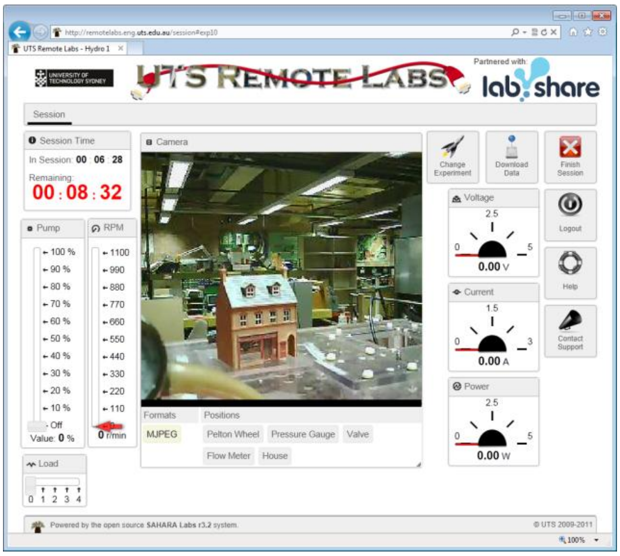


Figure 1: A typical remote laboratory system:
The UTS Hydroelectric apparatus allows students to explore hydro-electric power generation using a system containing water tanks, a pelton wheel and turbine, and a variable load.

A significant benefit that arises from the use of remote labs is the potential for greatly enhanced sharing of resources between institutions, with the resultant benefits regarding amortisation of costs and access to a more diverse range of lab facilities. The extent of sharing however will be influenced by the ease with which the sharing can be managed by the owners of the remote labs. Where the lab owner must individually manage each user, this will be a potential block to sharing. Whilst federated access technologies (where user identities across different domains can be linked) provide a partial solution, they still require users to visit the lab owner’s site in order to gain access to the labs managed by that owner. An alternative is to modify the remote lab management systems so that any user organisation can create their own aggregation of labs, where the labs can be drawn from any lab provider that has authorised their access. The user organisation can then take on the responsibility for managing the access by their own users across this collection.

This paper considers the feasibility and system design implications of supporting this richer form of federated access, whereby users can transparently access, through their own local system, a range of apparatus drawn from diverse external sources.

Shared Access to Remote Laboratories

Remote laboratories, whilst not a universal panacea to the challenges of developing and maintaining teaching laboratories, have been shown to have a number of potential benefits. Apart from the obvious benefit or providing students with increased flexibility with regard to time and location of access, pedagogic research has also shown that whilst some learning outcomes are much better met using hands-on laboratory access, there are other learning outcomes that are actually enhanced through remote access (Lindsay & Good, 2005). There has also been some limited research that has begun exploring the opportunities provided by the computer-mediated remote lab interface for supporting augmented reality that enriches the learning experiences (Lowe, Bharathy, Stumpers, & Yeung, 2012).

There has been a much greater emphasis on the potential for remote laboratories to support substantial sharing of laboratory facilities (Lowe, Conlon, et al., 2012) with the concomitant benefits in cost savings and access to more diverse facilities. As a consequence of this there have been many initiatives focused explicitly on facilitating cross-institutional sharing of remote laboratories. Examples include, but are not limited to: the NSF-funded World Wide Student Laboratory project (originating in the mid-1990's, though dormant since 2007) (Arodzero, 1998); LearNet (Rohrig & Bischoff, 2003), ProLearn (Gomes & Bogosyan, 2009), and PEMCWebLab (Bauer, Fedák, & Rompelman, 2008); and PEARL (Scanlon, Colwell, Cooper, & Di Paolo, 2004) (all run during the early to mid-2000's and involving consortia of Universities focused on sharing laboratories). More recent work focused on shared lab access has included projects such as: LiLa (Richter, Böhringer, & Jeschke, 2009); Labshare (Lowe, Conlon, et al., 2012); iSES (Schauer et al., 2005); NANSLO (Western Interstate Commission for Higher Education, 2012); and UniSchooLabs (Scienter, 2012).

The significant number of initiatives aiming to support shared access to remote labs illustrates the strength of the belief in the potential benefits. In most cases these sharing initiatives have been based around peer-to-peer sharing, where the developer/owner of a lab makes it available to others, based on some appropriate model that justifies the effort required to support the sharing. This in turn leads to a consideration of how such sharing can be appropriately supported.

Many extant remote labs have been initially developed as stand-along systems where students directly access the apparatus, potentially through a dedicated web interface. Whilst this enables access, it doesn't necessarily provide an effective mechanism for coordinating large-scale shared access. Indeed there are a range of support functionalities that become important once the usage of the lab increases. Examples include: user authentication and access authorisation; management of pools of labs; scheduling of access by users (including mechanisms such as management of access reservations and queues of users waiting for access); apparatus monitoring and testing; experiment results management; etc. (Yeung, Lowe, & Murray, 2010).

A number of Remote Laboratory Management Systems (RLMS) have been developed that provide various combinations of these functionalities. Examples include the iLabs Shared Architecture (ISA) (Harward et al., 2008), Sahara (Labshare, 2010), WebDeusto (Garcia-Zubia et al., 2009) and LiLa (Richter et al., 2009). Each of these systems is based on a different architecture but in general they have a similar purpose – to manage and support shared access to a collection of remote laboratories.

This shared access is typically achieved by establishing appropriate access credentials for users on the RLMS who then can then log in and request access to the apparatus. This request can be either on a first-available basis (where they would be placed in a queue) or by making a reservation for access at a particular time. Whilst these mechanisms support coordination of access they do typically require that user management to be on the lab provider's system. The effort required for this management can, in turn, be a significant disincentive to sharing.

One partial solution to this challenge is to leverage technologies that support federated access, such as shibboleth. Shibboleth is a software/networking middleware technology that supports distributed identity management and federated access to systems. Essentially a user requesting access to a "content provider" (in this case, the remote lab owner) is redirected to their identity provider (IdP) (in this case the remote lab user's home institution). This IdP authenticates the user and then vouchsafes the user to the content provider – provided the content provider trusts that IdP.

This approach simplifies the user management (since the owner of the remote lab need only provide a mechanism for confirming that a group of users from an external organisation are trusted and can have a specified level of access. Whilst this simplifies the sharing from the perspective of the lab provider, it is still somewhat of an inadequate solution from the perspective of the lab users.

Consider the situation where a given user must access a range of 5 different labs (potentially in different courses/subjects) that are hosted by 5 different institutions. The access locations and interfaces will be different for each lab. A preferred solution is for this user's institution to create their own local collection for these 5 remote laboratories, so that from the perspective of the users they are all accessed from one location and with a common log-in and access management infrastructure. Whilst the apparatus will be located at diverse remote distributed locations, this is essentially irrelevant to the user who accesses them through a single portal. It is the creation of such a solution that we consider in this paper.

The Sahara Remote Laboratory Architecture

In order to explore the feasibility, and architectural implications, of modifying a RLMS to support the creation of local collections of distributed remote laboratories we have elected to use the Sahara RLMS.

Sahara is an open-source software suite that was developed, in part, within the Labshare project (Lowe, Conlon, et al., 2012). Labshare was an Australian project formed in late 2008 that aimed to support the creation of a nationally shared network of remote laboratories in the context of Australian higher education institutions. The Labshare project resulted in the launch of The Labshare Institute (TLI) which continues to support remote laboratory sharing programs.

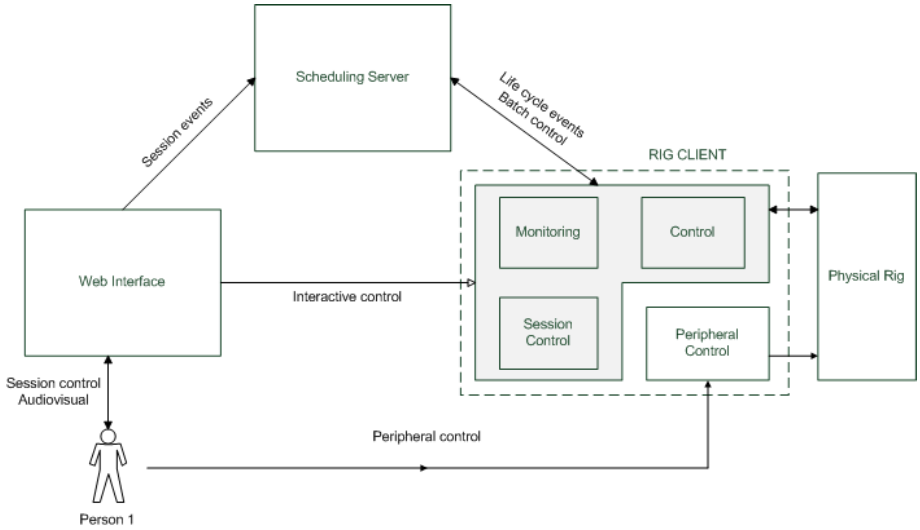


Figure 2. The architecture of the Sahara Remote Laboratory Management System (*from Sahara Development Handbook, available at http://sourceforge.net/projects/labshare-sahara/files/Documentation/, by permission of The University of Technology, Sydney*).

Sahara was a key outcome of the Labshare project and is one of the world's leading remote laboratory management systems. Sahara supports a rich suite of user and laboratories management functions, including the ability to monitor the health of labs, and to allow users to queue for and/or reserve laboratories. The latest public release of Sahara (R3.2) also incorporates support for federated user access using shibboleth via the Australian Access

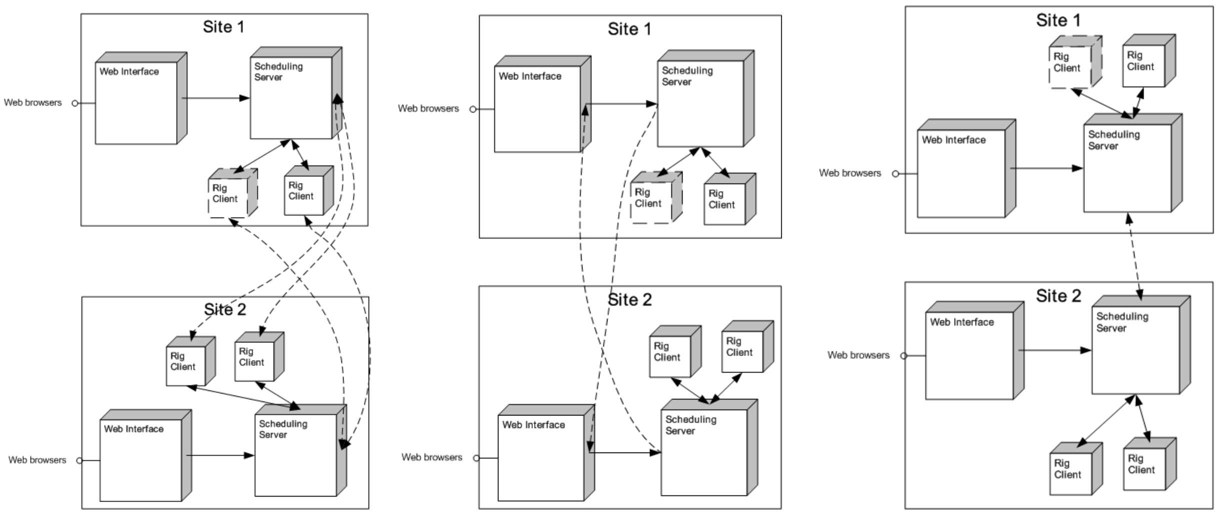
Federation. The basic architecture of Sahara is show in Figure 2. This contains the following key components: the web interface; the scheduling server; and the rig client.

The Web Interface is the user interface to SAHARA Labs. Its roles include authenticating a user, providing the interface for users to request access to a resource and providing the interface to control a rig. It provides programmatic interfaces to implement aspects such as how users are authenticated and presentation. It also include a suite of components for implementing the web interface associated with different rig types.

The Scheduling Server is the middleware component that manages the scheduling processes of the remote laboratory rigs, including tracking the state of rigs and assigning them to users based on either queued access or time-based reservations. It is also responsible for managing the running sessions according to the allocated times as well as logging all events and activities. The Scheduling Server is agnostic with regard to the specific design of a rig and relies on a rig client application to turn scheduling requests into rig specific behaviour.

The Rig Client provides a software abstraction of each rig and converts abstract requests from the scheduling server into rig specific actions. If a user is being assigned to the rig, its Rig Client provides the behaviour to actually allow the user to access the rig. The Rig Client provides a programmatic interface to allow this behaviour to be defined as a set of Java interfaces. The Rig Client manages the rig and sends status updates to the Scheduling Server. If the Rig Client is not operating, from the perspective of the Scheduling Server, the rig is not operational and no user will be assigned to it. The Rig Client also provides a control channel to directly interface with the rig or as an auxiliary to an external control program.

Sahara is currently in use by multiple universities, with each installation of Sahara as an isolated instance that manages a distinct collection of remote laboratories. There is no communication between Sahara instances and hence no direct sharing of laboratories between instances. This means that in order to gain access to a specific remote laboratory a user must log in to the Sahara server that manages that laboratory (albeit using their own home institution access account).



(a) (b) (c)

Figure 3: Example RLMS Federation Architectures: (a) Multiple control of rig clients; (b) Web interface communication; (c) Scheduling server communication

Federated System Design

There are a number of different possible ways in which the Sahara architecture can be extended to incorporate the desired functionality, where multiple rigs, managed by different distributed Sahara servers, can be aggregated into one local collection. A range of possibilities were considered, though we only discuss the four most illustrative examples here (due to space constraints):

Architecture 1: Federation with Centralised Scheduling Server

This possibility (not shown in Figure 3) involved a ‘federation’ built with a single centralised Scheduling Server and Web Interface that communicates with rig clients (and hence apparatus) that is distributed across multiple “provider” sites. Whilst simple to implement (indeed this could be implemented with no change to the existing architecture) this approach can be immediately excluded due to several critical flaws. Firstly, this precludes different organisations creating their own virtual collections (and managing their own users’ access to the rigs within these collections). The centralised server also forms a single point of failure and a potential performance bottleneck.

Architecture 2: Federation by multiple control of Rig Clients

Figure 3a illustrates an architecture that returns autonomy back to the individual remote laboratory sites – each of which maintains its own Sahara server that manages one or more remote laboratories (or indeed none, if this server is only used to support access to external labs). The extension to the current Sahara server is that the Scheduling server is able to communicate not only with its own rig client, but also rig clients at other sites. In this architecture the obvious challenge is the difficulty in ensuring the rig client is only in use by a single user at a time – and can only be reserved for use by a single user at a time. This implies a significant degree of increased complexity in the rig client, which would need to incorporate a degree of scheduling capability that is currently embedded in the scheduling server.

Architecture 3: Federation by Web Interface communication

Figure 3b shows an architecture in which communication between sites is through a users’ Sahara Server requesting access to a lab provider’s functionality through their Web Interface (in essence, the user’s Sahara server acts much as a local user would). This is a plausible solution since it recognises that it is the Web Interface that is the existing interface to the remote laboratory functionality and thus it proxies the Scheduling Server’s functionality through an interface mediated by login credentials of a user. An advantage of this approach is that it would require relatively little modification to the existing design. It has a significant disadvantage however with regard to the web interface becoming a potential performance bottleneck.

Architecture 4: Federation by Scheduling Server Communication

The architecture shown in Figure 3c involves direct communication between the Scheduling Servers within the Sahara systems. This recognises that it is the Scheduling Server which is the ultimate destination of functionality required for a federated remote laboratory system. If a user is requesting to use a rig it is the Scheduling Server that ultimately decides whether to queue the user and if and when they are assigned to a rig. The Scheduling Server already has a programmatic interface to provide this through a SOAP interface but this is inadequate for external consumption since it puts the onus of authentication on the Web Interface (only the authorisation decision is at the Scheduling Server). To implement this architecture would require a new federation service to be developed on the Scheduling Server that provides an interface to user level Scheduling Server operations such as queuing and session termination. It would also require extending the current Scheduling Server permission structure so there is a permission that resolves to calling another site's Scheduling Server federation interface to perform things like queuing or reservations of that site's resources.

This architecture is essentially a 'peer-to-peer' architecture with consumer and provider Scheduling Servers being peers of each other. The advantage of this architecture is the provider Sahara knows about the consumer Sahara and so there may be duplex communication between them. For example a consumer Sahara server can request to put a user into the queue on the provider Sahara server then wait for the provider to notify that the user has been assigned to a rig.

Evaluation

As outlined above, architectures 3 and 4 both represent feasible solutions. In order to assess their relative metrics we undertook a conceptual evaluation of both of them using software architecture quality attributes drawn from (Reekie & McAdam, 2006). For each attribute we considered which architecture best exhibits that attribute. The results of this evaluation are shown in Table 1.

Table 1: Architecture quality evaluations

Quality Attribute (Reekie & McAdam, 2006)	**Architecture 3 – Web Interface Communications**	**Architecture 4 – Scheduling Server Communications**	**Architectural preference**
Performance How much work a system can perform for a given resource.	Does not. The consumer is required to poll the provider.	The provider Scheduling Server can push updates to the consumer.	Strongly 4
Usability Consideration of the inter-links between software and the person.	Does not. Polling introduces a lag between an event occurring and the next poll.	Pushed statuses are temporal at the time an event occurs.	Strongly 4
Reliability How infrequently the system fails to perform its intended function.	Does not require as much state to be preserved between consumer and provider as it requires a consumer to pull through all details with a poll.	Requires state synchronisation between consumer and provider to correctly push updates.	Weakly 3
Security Ability to ensure that it can and will be used in the intended way.	Has the Web Interface as the authentication barrier using user authentication details.	Uses a customised interface specifically developed for federation consumption.	Strongly 4
Maintainability Ease with which the system can have defects corrected.	Reuses the existing Web Interface REST / JSON code which is production tested. Hard to upgrade as it is a PHP application with lots of separate files.	Requires new code to be written. Changes confined to Scheduling Server which as a Java application with installers that upgrade existing installations, it is straight forward to upgrade.	Weakly 4
Reusability Ease with which software elements can be used in multiple contexts.	The Web Interface REST / JSON interface may be used by other applications.	Does not.	Strongly 3
Scalability Ease with which the system can be scaled.	Does not.	Push updates eliminate unneeded poll communication.	Strongly 4.

Looking at the table of quality attributes, architecture 3 is the preferred architecture based on two attributes (reliability and reusability). In contrast, architecture 4 is the preferred architecture based on 5 attributes (performance, usability, security, maintainability, scalability).

Based on this evaluation architecture 4 was chosen to be implemented. The Sahara system was modified accordingly to produce a prototype version that incorporated the new functionality. Subsequently a test setup was established that involved four distinct Sahara instances:

- Labshare for Schools (LHS): This site managed the Hydro rig (shown in Figure 1) and was a consumer of the Inclined Plane Rig (IPR) from the PHYS Sahara server.
- UTS:Physics Remote Laboratory (PHYS): This site managed the IPR rig and was a consumer of the Hydro rig.
- Apollo (AP): This site managed no rigs, and is a consumer of the Hydro and IPR rigs.
- sahara-labs.ws (SLWS): This site was hosted on the Amazon EC2 cloud (physically on the United States east coast). It managed no rigs, but was a consumer of the IPR and Hydro rigs.

Scenario testing was then utilised to assess the behaviour and performance of the system. The scenarios that were tested included:

- Using a rig with direct assignment
- Using a rig with queuing assignment
- Using a rig with reservation assignment

Each of these scenarios was executed to access each of the rigs from each of the Sahara servers. With the exception of one problem associated with reservations having an incorrect status (arising from problems with translating different time zones between the linked Sahara servers) all tests passed and no significant issues were identified.

Conclusions

This paper has described the issues associated with distributed shared access to remote laboratories. Specifically, consideration was given to the feasibility of creating local collections of remote labs drawn from diverse remote labs providers, and hence creating a federated system where a lab located at a remote providers site could be successfully used by users logging into their own local remote lab system.

To achieve this, the Sahara Remote Lab Management System was discussed and four alternative architectures were explored. From these alternatives, a candidate architecture was selected and subsequently used to implement a proof-of-concept prototype system. Evaluation of this prototype clearly demonstrated that from the perspective of users, the system behaved essentially identically irrespective of whether the labs were managed by the Sahara server to which the users were logged into, or whether the labs were managed by a separate “provider” Sahara server.

Whilst not yet a production quality system ready for deployment, the prototype has clearly shown that such a system can be readily designed and implemented. Full deployment could be achieved by: undertaking a full test cycle and associated bug fixes; completion of a federation management interface to handle establishing the links between servers; implementation of transport layer security of the federation services; and consideration of how experimental results should be handled.

References

Arodzero, A. (1998). *World Wide Student Laboratory* (pp. 1-8). Eugene, Oregon, USA. Retrieved from http://lanl.arxiv.org/pdf/physics/9806044.pdf

Bauer, P., Fedák, V., & Rompelman, O. (2008). PEMCWebLab - Distance and Virtual Laboratories in Electrical Engineering – Development and Trends. *Power Electronics and Motion Control Conference* (pp. 2354-2359). Poznan, Poland: IEEE.

Garcia-Zubia, J., Orduña, P., Irurzun, J., Hernández, U., Sancristobal, E., Martín, S., Castro, M., et al. (2009). Towards an Extensible WebLab Architecture. *ICELIE '09. 3rd IEEE International Conference on E-Learning in Industrial Electronics* (pp. 115-120). Porto, Portugal: IEEE.

Gomes, L., & Bogosyan, S. (2009). Current Trends in Remote Laboratories. *Industrial Electronics, IEEE Transactions on*, *56*(12), 4744-4756. IEEE. doi:10.1109/TIE.2009.2033293

Harward, V. J., del Alamo, J. A., Lerman, S. R., Bailey, P. H., Carpenter, J., DeLong, K., Felknor, C., et al. (2008). The iLab shared architecture: A Web Services infrastructure to build communities of Internet accessible laboratories. *Proceedings of the IEEE*, *96*(6), 931.

Labshare. (2010). Sahara Labs. Retrieved June 3, 2010, from http://sourceforge.net/projects/labshare-sahara/

Lindsay, E., & Good, M. (2005). Effects of laboratory access modes upon learning outcomes. *Education, IEEE Transactions on*, *48*(4), 619-631.

Lowe, D., Bharathy, G., Stumpers, B., & Yeung, H. (2012). Laboratory Lesson Plans : Opportunities Created by Remote Laboratories. *REV2012 - Remote Engineering & Virtual Instrumentation* (p. In Press). Bilbao, Spain: IAOE.

Lowe, D., Conlon, S., Murray, S., Weber, L., Villefromoy, M. D. L., Lindsay, E., Nafalski, A., et al. (2012). LabShare: Towards Cross-Institutional Laboratory Sharing. In A. Azad, M. Auer, & J. Harward (Eds.), *Internet Accessible Remote Laboratories: Scalable E-Learning Tools for Engineering and Science Disciplines* (1st ed., pp. 453-467). Hershey, PA, USA: IGI Global.

Reekie, J., & McAdam, R. (2006). *A Software Architecture Primer*. Sydney: Angophora Press.

Richter, T., Böhringer, D., & Jeschke, S. (2009). LiLa: A European Project on Networked Experiments. *REV 2009: 6th International Conference on Remote Engineering and Virtual Instrumentation*. Bridgeport, USA: IAOE. Retrieved from http://eprints.physik.tu-berlin.de/304/

Rohrig, C., & Bischoff, A. (2003). Web-based Environment for Collaborative Remote Experimentation. *Proceedings of the 42nd IEEE Conference Decision and Control*. Maui, Hawaii, USA: IEEE.

Scanlon, E., Colwell, C., Cooper, M., & Di Paolo, T. (2004). Remote experiments, re-versioning and re-thinking science learning. *Computers & Education*, *43*(1-2), 153-163. doi:10.1016/j.compedu.2003.12.010

Schauer, F., Kuřitka, I., Lustig, F., Centre, P., Bata, T., Masaryk, S. T. G., Republic, C., et al. (2005). Creative Laboratory Experiments For Basic Physics Using Computer Data Collection And Evaluation Exemplified On The Intelligent School Experimental System (ISES). *iCEER'2005: Exploring Innovation in Education and Research: Exploring Innovation in Education and Research* (pp. 305-312). Taiwan: iCEER. Retrieved from http://www.ises.info/ises_data/publications/SchauerLustigArticle2004_23_0023.pdf

Scienter. (2012). UniSchooLabs. Retrieved April 4, 2012, from http://unischoolabs.eun.org

Western Interstate Commission for Higher Education. (2012). North American Network of Science Labs Online (NANSLO). Retrieved April 4, 2012, from http://www.wiche.edu/nanslo

Yeung, H., Lowe, D., & Murray, S. (2010). Interoperability of Remote Laboratories Systems. *International Journal of Online Engineering (iJOE)*, *6*(SI1), 71-80. doi:10.3991/ijoe.v6s1.1387

Copyright statement