Interdependence of Booking and Queuing in Remote Laboratory Scheduling

D. Lowe[1] and N. Orou[1]
[1] Centre for Real-time Information Networks
University of Technology, Sydney

Abstract— **Optimized scheduling can significantly improve the utilization level of Remote Access Laboratories (RAL) resources and decrease the waiting time. Current RAL systems have typically supported either booking or queuing, though rarely both. In this paper we investigate issues that arise when a single RAL resource (or pool of resources) supports both modes for gaining access. This research analyses the scheduling algorithm utilized by the Sahara RAL system in order to investigate any limitations that affect the system utilization. We identify a number of current issues and propose specific modifications to address them. The proposed modifications should lead to increased utilization and improved student experience, and hence widen the scope of RAL usage.**

Index Terms— **Remote access laboratories (RAL), scheduling system, RAL's resources utilization.**

I. INTRODUCTION

Laboratories are important tools for education and scientific research. Scarce resources such as funds, space, and staffing [1] and the rapid evolution of computer technology and communication networks have supported the emergence of Remote Access Laboratories (RALs) in the mid 1990's. Since then RALs have become increasingly sophisticated with their deployment increasing, possibly partly due to significant benefits such as flexibility of access [2], ability to share resources and labs [3, 4], security of users, data, and devices [3, 5], and accessibility for disadvantaged users [5, 6] among many others. However, challenges within RALs with regard to software architectures as well as the priorities of the provided features have also merged. Considering flexibility, as one of the most important features of RALs, where students can access and do experiments any time and from anywhere, an important challenge is of finding the optimum access methods or scheduling algorithms.

RALs, in general, can be classified into two main types; interactive and non-interactive „or batch" laboratories. With regard to „interactive RALs" type of, scheduling refers to planning the times at which certain events/ activities should take place- particularly those related to access by users. A scheduling system needs to decide how to assign the RAL resources between these events/ activities and also defines the RAL system" actions/ reactions. These could take place previous to, during, or after the occurrence of these events/ activities. Events can refer to students" requests to access the RAL to perform an experiment, cancelling such requests, or finishing/quitting an experiment. Activities may represent an actual usage „session" during which a „rig" is used by a student, as well as other events such as maintenance outages performed by the RAL system.

While queuing is mostly used in the batch RALs, interactive RALs may use a range of different scheduling methods, most typically queuing, booking, or a hybrid scheme. The choice (and performance) of each scheme is likely to depend on a number of factors including; the number of concurrent users, the available rigs, and the usage duration of each user [7]. In [8] QoS and Service Level Agreements have been also considered as significant aspects that influence the choice of access method for RALs.

In this paper we will consider the issue of scheduling within the context of RALs. Our consideration has a particular focus on the effect of the implemented scheduling scheme, and specifically, the implication of using more than one scheduling method within a RAL on the utilization level of the RAL resources. To assess this issue we consider Sahara platform (explained in detail in section III). This platform was developed as part of the LabShare project [9]. Current version supports two types of scheduling methods; priority queues and calendar-based booking [7].

The remainder of the paper is organized as follows: in section II a review of scheduling algorithms, specifically, in RAL systems is presented. Section III introduces the Sahara RAL system and discusses its scheduling system. In section IV we present an actual example of the lab usage within a specific semester and analyze it in terms of the utilization of the lab resources. Section V contains the discussion of the results and suggestions for algorithm modifications. Finally we present the conclusions and discuss future work in section VI.

II. BACKGROUND

The concept of resource scheduling is applied across many fields of our daily life, including areas such as transport, hotels reservation, hospitals or private clinics" appointments, and online shopping. For instance, if a person wants to take a conveyance at some arbitrary time, then they may feel uneasy waiting for the next available conveyance if the departure intervals are long. Therefore, providing time schedules for all conveyances will help passengers to plan their travel with ease in advance and avoiding wasting time through waiting. In situations where either the resource availability or the resource demand is less predictable, waiting may become

N. Orou wishes to acknowledge the support of The Ministry of Higher Education and Scientific Research in Iraq for supporting her doctoral studies.

978-1-4673-2542-4/12/$31.00 ©2012 IEEE

inevitable, even though there are various approaches available to estimate the waiting time. For example, many call centers (e.g. Dell‟s support website) provides information on the length of the waiting list and an estimate for the waiting time when a customer cannot get immediate assistance [10]. In all of these cases, scheduling is important, providing a way of accessing a system, using its resources, and managing the periods of this usage.

A number of researchers have considered different approaches to the scheduling of access to limited resources. In the early work of [11], a scheduling function was used as the mechanism to manage access and use of a distributed system‟s resources by different users. In this sense, the efficiency and effectiveness of a scheduling function is based on satisfying the users‟ needs regarding the required system resources as well as the mechanism to reach them. Their hierarchical taxonomy for the resource management problem within distributed computing systems (DCS) attempted to provide a common terminology and classification mechanism that is necessary in addressing this problem. In the same context of DCS, another taxonomy concerning the dynamic task scheduling issue was presented [12].

Educational laboratories, even traditional hands-on laboratories, require some kind of scheduling to manage access to their resources and the sessions‟ timetables. For example, students may be able to sign up for available blocks of time where they have sole access to the required equipment.

Within the more specific field of RALs, it is only over the last few years that research has begun to address the scheduling issue. As RALs combine both distributed systems and laboratories, scheduling schemes and resource allocation algorithms gain more importance (or become more crucial). An effective approach to scheduling can provide increased opportunities to the students, specifically, in terms of ease and flexibility of access to the RAL.

Numerous approaches to scheduling have been considered, often drawing from concepts in related domains. For example, some research has considered scheduling and managing access to RAL as another resource within the Grid [8] where approaches based on service level agreements and advanced reservation are used to access the network resources. Others researchers have presented a “Web Service-based MetaScheduling Service” to coordinate the allocation of timeslots with local resources in Grid-Computing environments [13].

Many researchers have shown that scheduling systems within RALs are of high diversity. In [7], a number of the most popular RALs have been analyzed in terms of their approach to scheduling. Various scheduling schemes were identified such as queuing, priority queuing, booking, shared resources booking, and imposed group booking. Conversely, many RALs have no scheduling system, with the RAL essentially consisting of a single item of apparatus with no system mediation of access (effectively whoever accesses the equipment first typically has control) [7]. Even where a RAL with a single item of apparatus does include a scheduling mechanism, this is often integrated directly with the apparatus (see, for example, VISIR [14]) [7, 15]

It is worth noting however that, along with other generic functionality such as user authentication and authorization, scheduling does not depend on the nature of the experiment (beyond the basic distinction between batch and interactive experiments [7]). For this reason, a number of the RAL systems that have been designed to manage multiple items of apparatus incorporate the scheduling algorithms as part of the tools provided by a rig management framework [16] rather than being built from scratch along with each item of apparatus.

In this context, the scheduling scheme could be an extension to the functionality of a Learning Management System (LMS) such as the use of Moodle in the MARVEL project [17, 18]. Alternatively, it could be built as an independent solution such as with the LiLa booking system [15, 19]. Other researchers (for example, [20]) have considered scheduling within the context of accessing RALs resources from within a group (federation) of RALs [21]. Within a federation of RALs, certain modifications should take place to facilitate access among different partners or a metadata set could be introduced to manage this issue [22].

There has also been research that focuses on analyzing a specific scheduling scheme and subsequently highlighting specific problem and suggesting proposed solutions [10], or presenting a novel scheduling scheme [23], [24].

One analysis of multiple schemes [25] argued that the hybrid or mixed-mode scheme, namely “slotted queuing mode” which combines the queuing and booking modes, is the best way to overcome the shortcomings of each of the other two schemes as well as to optimize the RAL resource utilization. The author described an implementation of the slotted queuing algorithm in NETLab [23]. However, there was no clear explanation of how this method improved the lab utilization.

Despite the emerging research on RAL scheduling there has not, to date, been a detailed analysis of how a specific implemented scheduling scheme in a RAL can affect the utilization level of the RAL. Understanding this issue in more detail is an important step in improving RAL flexibility and cost.

III. LABSHARE SAHARA RAL

A. Background

The earliest RAL software system of the University of Technology Sydney (UTS) was developed in 2000-2005. This system, which retrospectively has come to be referred to as Sahara release 1 [9], was adopted as the base of the much broader Labshare project. This project was a joint initiative of the Universities belonging to the Australian Technology Network (UTS, Curtin, UniSA, RMIT and QUT). LabShare aimed to “establish a national approach to the sharing of remote laboratories that will provide higher quality RALs with greater student flexibility, improved educational outcomes, improved financial sustainability, and enhanced scalability in terms of coping with high loads of students” (see http://www.labshare.edu.au). To satisfy each of these goals, Sahara has undergone a major redesign process producing new revisions of Sahara (release 2, 3 and onwards) [9]. A final outcome of the LabShare project was the recent creation of The LabShare Institute (TLI) as a not-for-profit organization that will be an independent

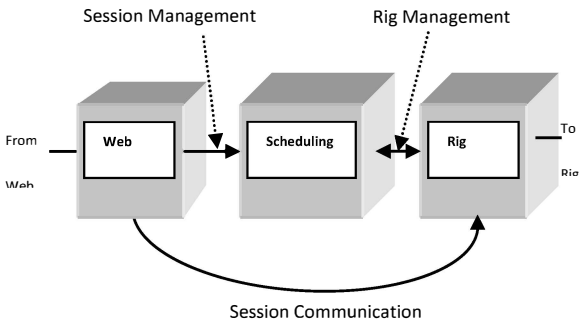

Figure 1: Sahara Current Architecture

service broker promoting, maintaining and even hosting remote laboratories [16].

B. *Architecture*

The underlying software architecture (Sahara) for the UTS remote laboratories is shown in Figure 1. This architecture supports two modes of experiment control: „interactive" and „peripheral" control[1]. Interactive control is achieved via the following basic software components:

- The Web Interface: through which students are authenticated and then select the apparatus (rig) they want to work on.
- The Scheduling Server: is the middleware that manages the scheduling process of the remote laboratory by tracking the state of rigs and assigns them to users. It is responsible for managing the running sessions according to the permitted allocated time. It also logs all events and activities.
- The Rig Client: This is the software component that provides a software abstraction of a rig and converts abstract requests to rig specific actions.

C. *Scheduling*

From release 3 onwards, the Sahara system supports two scheduling methods to handle the user requests: queuing and booking.

1) *Priority queuing:*

Queuing supports „on demand" requests, providing "soonest available" access to a selected rig (or collection of rigs). The basic logic for the queuing process within Sahara is as follows:

a. A user attempts to log into the Sahara server, is authenticated and is then provided with a list of the rigs for which they have authorization;

b. The user selects a particular rig (or pool of rigs) and then is presented with the current status of the rig and given the option to queue for access, as in Figure 2;

c. When a user makes a request to access a rig (or any one of a collection of rigs) the request is placed in the queue, with associated information indicating which rig (or collection of rigs) they have requested access

[1] Documentation on the Sahara architecture, and in particular the design of rig clients, can be found at: http://sourceforge.net/projects/labshare-sahara/files/Documentation/

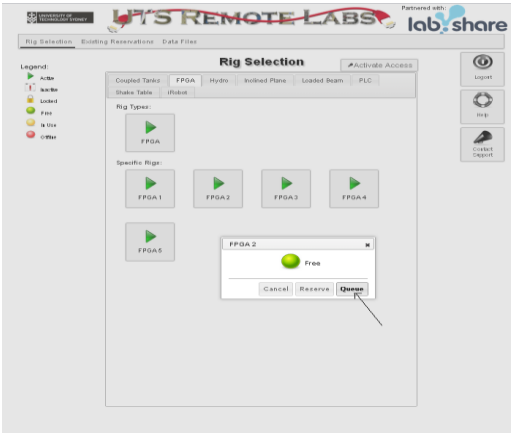

Figure 2: User selects to queue a specific rig (http://remotelabs.eng.uts.edu.au/queue)

to, the default usage time specified for their user class, and any other relevant parameters;

d. When any rig becomes available, Sahara will scan through the queue looking for the first request from amongst those that are the highest priority, and which satisfies the following criteria:

   i. The request is either for the available rig, or for a pool which contains the available rig; and

   ii. The request can be completed prior to the next booking for this resource (i.e. the next booked session for this resource is no sooner than the current time plus the default usage time for this user on this rig);

   If there is a conflicting booking and if that booking is for one of a pool of rigs, Sahara will attempt to move the booking to a different rig in the pool so that there is time available for the request.

e. Whilst a user has a pending request which is in the queue, they will be presented with a screen indicating how long they have been waiting in the queue.

f. Once a user is allocated to a rig, the rig page will appear and the experiment session will be started, including a timer counting the session time defined by the system, (as shown in Figure 3).

2) *Reservation (Calendar Booking)*

This form of access request allows users to make a reservation that provides guaranteed access to a rig (or one of a pool or rigs) at a specified time. In this process steps a-b of the queuing described above are the same except

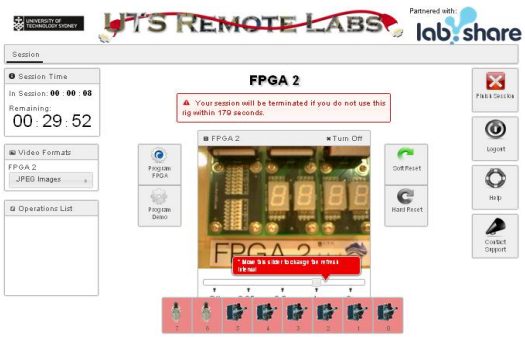

Figure 3: The selected rig session starts (http://remotelabs.eng.uts.edu.au/session/index)

Figure 4: The rig reservation page
http://remotelabs.eng.uts.edu.au/bookings/index/pid/36

that the user selects the (reserve option) rather than the (queue option), the process then proceeds as follows;

c. The rig reservation page will appear with the available time slots that can be booked (colored in green) and showing the maximum allowed reservation time for the selected rig, see Figure 4;

d. Once a time slot is selected, the reservation is created, a confirmation message appears, and an email message is sent to the user with the reservation details. The rig reservation page will show the reserved time slots (colored in grey) for all future accesses.

In both types of scheduling, once a user is allocated to a rig (i.e. a session has commenced) the session is monitored. The session is considered to have ended when: the user explicitly terminates their session; the user fails to commence the session or is inactive for a given timeout period (typically 10 minutes); the allowed session time is completed and no further extensions are allowed (a session can usually be extended a finite number of times if there is no pending user who wishes to use the rig).

The above descriptions imply a number of dependencies between the queuing and booking scheduling algorithms. For example, consider the case of a user in a queue waiting for access to a specific rig. The user cannot be allocated to the rig even if it currently available unless their session (at maximum length allowed) is guaranteed to be complete before any pending booking comes due. This interdependence can therefore, potentially, lead to time periods where a rig is idle even though it is currently the subject of usage demands. An analysis of how serious an issue this is will be presented in Section IV.

IV. Analysis

A. Process

To assess the extent to which booking and scheduling algorithms interact, and to determine possible ways of improving utilization levels, we undertook an analysis of existing rig allocation data. UTS hosts a set of coupled tank rigs which are used to undertake experiments in systems control (essentially the students undertake open-loop modeling of the equipment, and then determine appropriate parameters for a PID controller).

During Spring semester 2011(Spring semester runs from August to November and is one to the two main teaching semesters at most Australian Universities) a cohort of students from another Australian University made use of two of the coupled tank rigs. The primary usage period ran from early September through to the middle of October, with the peak usage commencing around October 1st and running for around 11 days. The students could choose to access either of the two rigs, and the scheduling could be via either queuing for access or by making a booking.

Extracts from the UTS Sahara system were obtained and analysed in Matlab. The data obtained included:

1- A listing of all bookings that were made for the relevant rigs and, where the booking was not cancelled, the rig session that corresponded to the redemption of this booking;

2- A list of all sessions on the rigs, including which rig was accessed, when the session was requested, when it commenced, when it ended, and why the session was terminated (by user request, due to an idle timeout, or for some other reason); and

3- A set of information on the rig types, user classes, permissions, rig capabilities, etc.

The Matlab code extracted relevant information from the database logs, analysed the results to determine which rigs were used and how they were used, and then plotted the results.

B. Results

Figure 5 shows a screendump from the rig scheduling analysis. This figure contains the data for both rigs that were in use during the period under analysis. The top graph provides a broad view of the overall usage of the *Coupled Tanks 3* rig. From this it can be readily seen that for most of the time there was no queue, and even during heavy usage the queue rarely exceeded 2-3 users.

The bottom graph provides a more detailed view of specific period of approximately 19 hours (extending from hour 981 to hour 1000, corresponding to approximately 10:00am on 1/10/2011 to 5am on 2/10/2011). A study of this figure shows some interesting behaviors emerging from the interplay between the booking and scheduling algorithms. Tracing through the time period we can see the following events:

982.5: The rig is in use by a student (#1), who had queued for access. The queue is initially empty, but around 983 another student arrives (#2) and queues for access.

983.6 Student #1 completes. Student #2 is still in the queue, but is not allocated the rig because there is a pending booking in 24 minutes (at 984). The student therefore remains in the queue.

Slightly later

986.2 By this time, student #2 is still in the queue due to a sequence of booked sessions. Several more students arrive (#3 and #4) around 986.5 and the queue grows in length to 3.

987 The current student completes their session, but the queued students are not allocated the rig as there is not sufficient time before the next booking. Over the next 3 hours (until 990) a series of booked sessions are allocated and then quickly ended due to time outs. i.e. the students who made the bookings did not redeem those bookings. The unredeemed bookings however did mean that the queued students could not access the rig, and it spent considerable time sitting idle.

990 Finally, at 990, the rig is again allocated to a booked session, which again times out (after 10 minutes). At this point the next booking is not until 991.5, and so finally a student in the queue

can be allocated (after waiting over 7 hours, despite 4.5 hours of unused rig time during that period). Not unsurprisingly, the student is not available and their session quickly times out, as does the session for the next student in the queue.

This sequence of events highlights a key problem with the current algorithms used for managing the interplay between the booking and queuing algorithms. In particular, there may potentially be significant proportions of time (especially during heavy usage periods) where there are students queued for use, but the rig is idle waiting for a pending booking.

To assess how significant a problem this is we analysed the peak usage periods during the timeline shown in Figure 5. This showed the following results for the 4 day period from hour 968 to hour 1064:

Total time:	92.00 hours
In use or pending allocation:	74.90 hours
Active use:	46.64 hours (62% of usage time)
Idle use:	9.18 hours (12% of usage time)
Waiting:	19.08 hours (26% of usage time)

In other words, there was 74.90 hours during which the rig was either allocated to a user, or there was a queue that wanted allocation but the rig was waiting for a pending booking. Of this time, around 9 hours (12%) was spent allocated to user who was not actually using the rig (in this case the system will, after 10 minutes, time out the user, and free up the rig). More surprisingly, there was 19 hours (26% of the time) that the rig was not allocated to anyone, but there were queued users waiting who could not be allocated because of a pending booking.

C. *Discussion*

The above case study highlights an unanticipated consequence of the interplay between booking and queuing schemes. More importantly, it demonstrates that considerable care must be taken in the design of scheduling algorithms if we are to optimize performance measures of RAL systems, including aspects such as overall level of utilization and queue waiting times.

We are currently investigating the impacts, initially through detailed simulations using the live request data from previous usage periods, of several potential modifications to the algorithms. These include:

a. *Movable booking times:* This involves modifying bookings so that they represent an approximate time, rather than an absolute time (much like a booking to see a doctor, where you arrive at the booked time, and may need to then wait a „short" period). Consider an example: there are two 1-hour bookings at 3:00pm and 4:00pm, and one user waiting in the queue. The user with the 3:00pm booking does not log in and so the session times out at 3:10pm. This however leaves only 50 minutes till the next booking and so the queued user (who is entitled to 60 minutes) will not be allocated. However, if we can move the 4:00pm booking back to 4:10pm then the queued use can be allocated (and indeed they are likely to complete the experiment in less time than

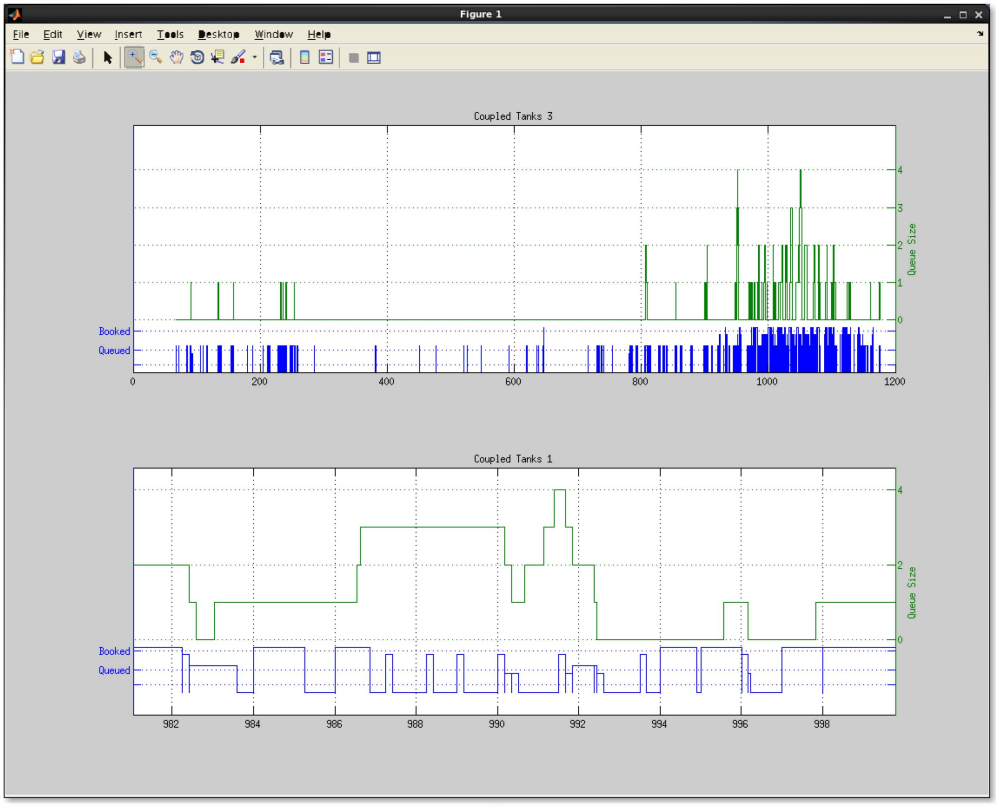

Figure 5: Rig usage analysis
(In each of the two graphs, the top curve shows the queue length over time and the bottom curve shows the status of the rig – specifically whether it is idle, in use by a user who queue, or in use by a user who made a booking. Where the rig status line is above the dotted line, this indicates that the rig was actively used. Where the status line is below the dotted line it indicates the session was allocated but the rig was not used during that session and hence the session timed out. The time axis is measured in hours).

the maximum allowed, and the 4:00pm booking may not end up being affected).

b. *Options for shorter allocations*: In this case, when a shorter time window becomes available (the 50 minute window in the above example) the users in the queue, who are nominally entitled to a longer period, are asked whether they would like to accept this shorter period. In many cases they may know that they are able to complete the experiment quickly and would be willing to accept a shorter timeslot if one is available, in preference to waiting for a slot that guarantees their maximum time allocation.

V. Conclusions

In this paper we have considered scheduling schemes for managing access to limited RAL resources. In particular we have shown that there can be complex interdependencies between different schemes that have adverse effects when they are combined in a single system. We utilized a case study of actual usage to demonstrate the problems that can arise, and proposed a number of changes that may potentially lead to improved allocation performance.

Acknowledgment

The authors wish to acknowledge the assistance of Michel de la Villefromoy and Michael Diponio with obtaining and understanding the Sahara log files.

References

[1] Q. Yuliang, L. Guo-Ping, Z. Geng, and H. Wenshan, "NCSLab: A Web-Based Global-Scale Control Laboratory With Rich Interactive Features," *IEEE Transactions on Industrial Electronics,* vol. 57, pp. 3253-3265, 2010.

[2] S. Paladini, J. B. da Silva, G. R. Alves, B. R. Fischer, and J. B. da Mota Alves, "Using Remote Lab Networks to Provide Support to Public Secondary School Education Level," in *11th IEEE International Conference on Computational Science and Engineering Workshops, 2008. CSEWORKSHOPS '08.* , 2008, pp. 275-280.

[3] G. Carnevali and G. Buttazzo, "A virtual laboratory environment for real-time experiments," in *Proceedings of the fifth IFAC International Symposium on Intelligent Components and Instruments for Control Applications, SICICA 2003*, Aveiro, Portugal, 2003, pp. 39-44.

[4] S. Murray, D. Lowe, E. Lindsay, V. Lasky, and D. Liu, "Experiences with a hybrid architecture for remote laboratories," in *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual*, 2008, pp. F1B-15-F1B-19.

[5] C. Gravier, J. Fayolle, B. Bayard, M. Ates, and J. Lardon, "State of the art about remote laboratories paradigms-foundations of ongoing mutations," *International Journal of Online Engineering i-JOE,* 2008.

[6] L. Gomes and S. Bogosyan, "Current Trends in Remote Laboratories," *IEEE Transactions on Industrial Electronics,* vol. 56, pp. 4744-4756, 2009.

[7] J. García-Zubia and P. Orduña, "Scheduling schemes among Internet Laboratories ecosystems " in *Remote Engineering & Virtual Instrumentation, REV*, 2011.

[8] P. Wieder, O. Waldrich, and W. Ziegler, "Advanced Techniques for Scheduling, Reservation, and Access Management for Remote Laboratories," in *e-Science and Grid Computing, 2006. e-Science '06. Second IEEE International Conference on*, 2006, pp. 128-128.

[9] H. Yeung, D. Lowe, and S. Murray, "Interoperability of Remote Laboratories Systems," *International Journal of Online Engineering (iJOE),* vol. 6, pp. pp. 71-80, 2010.

[10] L. Yaoye, S. K. Esche, and C. Chassapis, "A scheduling system for shared online laboratory resources," in *Frontiers in Education Conf, 2008. FIE 2008. 38th Annual*, 2008, pp. T2B-1-T2B-6.

[11] T. C. Casavant and J. G. Kuhl, "A Taxonomy of Scheduling in General-Purpose Dist Computing Systems," *IEEE Transactions on software engineering,* vol. 14, February 1988.

[12] H. G. Rotithor, "Taxonomy of dynamic task scheduling schemes in distributed computing system," in *Computers and Digital Techniques, IEE Proceedings*, 1994.

[13] A. Gallardo, T. Richter, P. Debicki, L. Bellido, V. Mateos, and V. Villagra, "A rig booking system for on-line laboratories," in *Global Engineering Education Conference (EDUCON), 2011 IEEE*, 2011, pp. 643-648.

[14] I. Gustavsson, J. Zackrisson, L. Håkansson, I. Claesson, T. Lagö, and "The VISIR project – an Open Source Software Initiative for Distributed Online Laboratories " in *Remote Engineering & Virtual Instrumentation, REV*, University of Porto, Portugal, 2007.

[15] U. Hernández-Jayo and J. García-Zubía, "A Remote and Reconfigurable Analog Electronics Laboratory based on IVI an LXI Technologies," in *Remote Engineering & Virtual Instrumentation, REV*, Brasov, Romania, 2011.

[16] P. Orduna, J. Irurzun, L. Rodriguez-Gil, J. Garcia-Zubia, and D. Lopez, "Reusing requirements among remote experiments for their development and integration under WebLab-Deusto," in *8th International Conference on Remote Engineering and Virtual Instrumentation*, Brasov, Romania, 2011.

[17] M. A. Bochicchio and A. Longo, "Extending LMS with Collaborative Remote Lab Features," in *Advanced Learning Technologies (ICALT), 2010 IEEE 10th International Conference on*, 2010, pp. 310-314.

[18] J. M. Ferreira and A. M. Cardoso, "A Moodle extension to book online labs," presented at the Remote Engineering & Virtual Instrumentation, REV, University Transilvania Brasov, Romania, 2005.

[19] A. Gallardo, V. Mateos, T. Richter, L. Bellido, P. Debicki, and D. 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, 2011.

[20] V. Kondabathini, S. Boutamina, and S. K. D. Vinjarapu, "A Theme to Unite The Resources of Different Remote Laboratories," in *2011 IEEE International Conference on Technology for Education (T4E)*, 2011, pp. 51-55.

[21] M. Diponio, " Investigation and Implementation of a Federated Remote Laboratory Architecture Using the Open Source Sahara Labs System," University of Technology, Sydney, Faculty of Engineering and Information Technology2011.

[22] P. P. Grube, D. Boehringer, T. Richter, C. Spiecker, N. Natho, C. Maier, and D. Zutin, "A metadata model for online laboratories," in *Global Engineering Education Conference (EDUCON), 2011 IEEE*, 2011, pp. 618-622.

[23] A. Maiti, "A hybrid algorithm for time scheduling in remotely triggered online laboratories," in *2011 IEEE on Global Engineering Education Conference (EDUCON)*, 2011, pp. 921-926.

[24] H. H. Saliah, L. Villardier, C. Kedowide, B. Assogba, and T. Wong, "Resource management strategies for remote virtual laboratory experimentation," in *Frontiers in Education Conference, 2000. FIE 2000. 30th Annual*, 2000, pp. T1D/8-T1D12 vol.1.

[25] A. Maiti, "Time Scheduling Schemes in Online Laboratory Management Systems," *International Journal of Online Engineering (iJOE),* vol. 6, 2010.

Authors

David Lowe is with the Centre for Real-time Information Networks, Faculty of Engineering and Information Technology, University of Technology, Sydney; 15 Broadway Ultimo, NSW 2007, Australia; (E-mail:david.lowe@uts.edu.au).

Nagham Orou is with the Centre for Real-time Information Networks, Faculty of Engineering and Information Technology, University of Technology, Sydney; 15 Broadway Ultimo, NSW 2007, Australia; (E-mail:Nagham.E.Orou@student.uts.edu.au).