RTLabOS Phase I: Software Infrastructure for Smart Grid Labs - Summary and Recommendations

RTLabOS D4.1

Heussen, Kai

Publication date:
2014

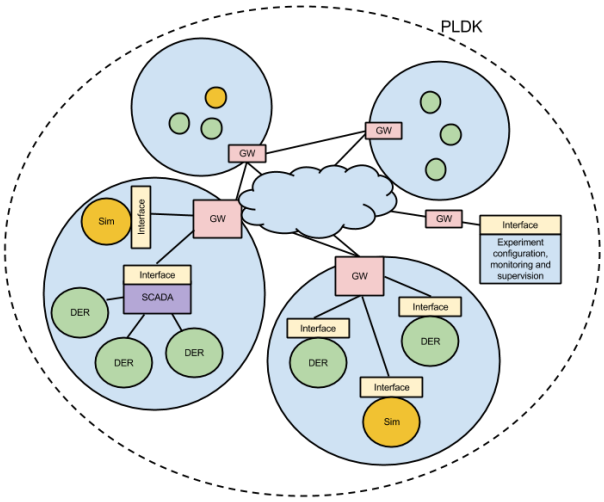
Document Version
Publisher's PDF, also known as Version of record

DTU Electrical Engineering

RTLabOS Phase I: Software Infrastructure for Smart Grid Labs

Summary and Recommendations

RTLabOS D4.1



Kai Heussen

November 2014

RTLabOS Phase I: Software Infrastructure for Smart Grid Labs
Summary and Recommendations

Report RTLabOS Phase I: D4.1

2014

By
Kai Heussen

Cover illustration: Anna Magdalena Kosek
Published by: Department of Electrical Engineering, Akademivej
Request report from: www.dtu.dk

Preface

The project RTLabOS: Phase I was conceived in 2012 as a small project to see how DTU and Spirae could explore the testing and demonstration of Spirae's platform and control system in PowerLabDK labs at two DTU sites in Lyngby and Risø. Here is a short story of this project. Demonstrating such a commercial grade control system was new for our labs. Not far along into the project, it became also clear that our ambition to facilitate and streamline such processes hit a nerve in the context smart grid labs. The questions we raised about facilitated development, accelerated deployment and testing of control software stimulated interest in the research community, and soon an interest group was found to join our quest, participating in workshops, supporting our survey, and collaborating on feasibility studies on lab deployments and co-simulation.
The eight-hour time difference between Spirae's development team and DTU's labs, was a challenge, but it served as excellent incentive to try out and optimize capabilities for remote deployment and testing at PowerLabDK. As usual with firsts, worked in the lab as it was intended on paper, but small setbacks generated learning opportunities in an overall rapid progress. As another first, we proved that the flexibility of SYSLAB software and developers can be asset also in the Lyngby labs.
After this first practical effort, more ideas were turned into practical feasibility studies, all recorded with learnings as examples and evidence for following up. Not all ideas could be tried in practice, but by carefully and systematically formulating use cases, we painted the larger picture. Of course, we are convinced of our ideas, and yet their practical significance could be very low; thankfully, by conducting workshops and surveys with other labs, and with local lab users, we have some understanding of who might actually care to see further development. This is what we share with you here, and we invite you to reflect and take it further.

In the first place I want to thank my colleagues from the core team, Anders, Anna and Oliver (DTU), as well as Holger (Spirae) for the energy and ideas they poured into this project. I'd also like to thank Henrik who had our back all along, and the many direct collaborators and supporters from DTU, Evgenia, Junjie, Nils, Hugo, Guangya, Per, … and from Spirae, Mahesh. Of course there have been many more, in particular the survey participants who took a strong interest in our work, and others involved as collaborators, co-authors, administrative supporters, workshop participants, etc. to all of you I am grateful for your encouraging participation.

Kgs. Lyngby, November 2014

Kai Heussen

Assistant Professor
Energy Systems Operation and Management
Center for Electric Power and Energy
DTU Electrical Engineering

Content

1. Introduction and Overview of Results

The project “RTLabOS Phase I” aims to provide a foundation for the further strategic development of the software infrastructure of PowerLabDK (PLDK, www.powerlab.dk) to support education, research and commercial activities. With the lifecycle of control software in view, we developed concepts and generated new experience for development, deployment and demonstration, toward integrating simulation and physical lab environments.

The requirements for laboratories in a Smart Grid context are moving toward further integrated systems, where the complexity and software intensity of technologies is increasing. While several research laboratories have experience with integrated experiments on development and demonstration of control concepts, the professional and research scope is widening to more integrated systems development and testing. A cornerstone for effective work across several development phases is the software infrastructure to operate the lab and support key activities.

This report presents an overview of the RTLabOS project and summarizes the key results. As illustrated in Figure 1, the main components of the project have been:

- the generation of development ideas toward a ***vision*** for the smart grid laboratory and its software infrastructure, through workshops, and surveying the state of the art
- to identify the actual needs of ***lab users*** in the PowerLabDK context
- the formulation of development ideas into structured ***software requirements*** by formulating use cases for supporting lab related activities
- to test development ideas in practice as ***feasibility studies***, to gain experience with concrete system tests, controller deployment and interface development

Highlights from these activities are shortly summarized in the following sections; for the full detail, please refer to the individual reports [1, 2, 3, 4, 5, 6, 7]

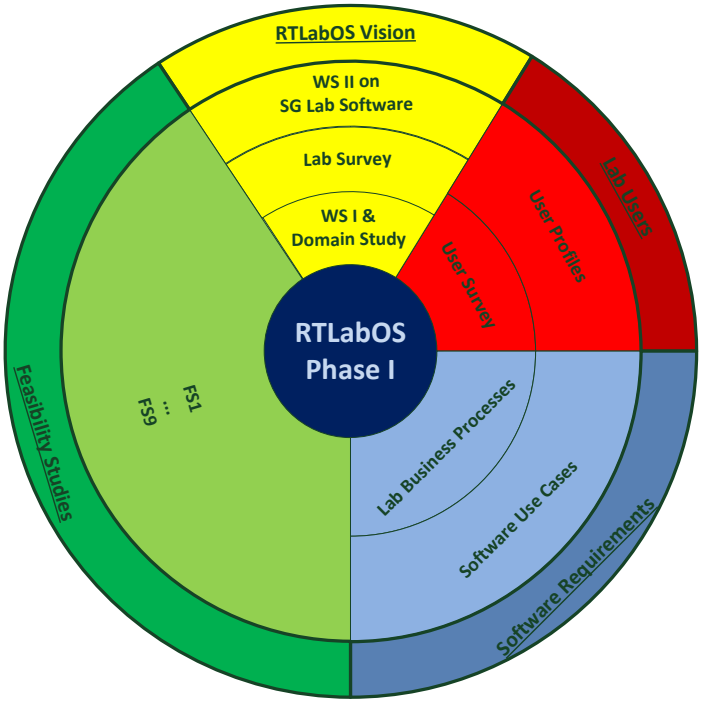


Figure 1 Illustration of RTLabOS Phase I project elements and relative efforts.

At the start of RTLabOS Phase I, a vision for was formulated to define a direction for the future smart grid lab software infrastructure:

> *The vision of RTLabOS is a supportive, real-time, cross-location laboratory software infrastructure for development and testing of topology independent and system-wide controls. Such an infrastructure will allow for seamless integration of simulated and physical components, and support open-platform- and standards-oriented development of solutions that can easily be deployed in the real world, enabling simulation and experiments with all relevant time-scales.*
>
> *Designed with all phases of experimental development in mind, it will offer support starting from experimental setup and configuration, through online supervision and monitoring, to the tracking of relevant data-sets from various sources. It will be based on a software architecture that strikes the balance between ease of access, meaning low-entry threshold and simple configuration, and the flexibility needed for laboratory software which is under constant development.*

The mission for RTLabOS Phase I then was defined as:

> *RTLabOS Phase I aims at assessing the state of the art, identifying requirements for a future lab environments meeting the vision as well as assessing potential software architecture.*

With this mission, the following aspects were to be considered to be in scope for the state of the art: *Interoperability; SCADA Systems & Lab Integration Platforms; Service-based Architecture in the automation fields; Lab Use Cases; Lab Facilities*.

The complete set of RTLabOS Phase I reports is [1-7]:

- D1.1 Domain Study
- D1.2 Lab Survey “State of the Art Smart Grid Laboratories”
- D2.1 Use Cases “Use Cases for Laboratory Software Infrastructure”
- D2.2 User Survey “Survey and Characterization of User Profiles and User Requirements”
- D3 Feasibility Studies “RTLabOS Feasibility Studies”
- D4.1 Final Report “RTLabOS Summary and Recommendations” *(this report)*
- D4.2 Dissemination Activities “RTLabOS Dissemination Activities”

All reports are publically available and retrievable from www.dtu.dk

1.1 Evolving the RTLabOS Vision: Workshops & Lab Survey

The keywords stated in the general vision give an impression of the high-level intentions for what a RTLabOS integration platform may offer; however, the vision offers no concrete field for which a state of the art could be established. To this end, state of the art and development ideas have been sketched and prioritized in several iterative steps: internally and through workshops with international participation. Summaries of the workshops are found in [7]. The state of the art is presented via a Domain Study [1], outlining the lab activities to be supported and related software categories, and a lab survey [2], providing anecdotal evidence of lab focus areas and software use.

Participants in the first two workshop (WS1 & WS2) have been internal from the RTLabOS project (Spirae & DTU), from PLDK and CEE (Centre for Electric Power and Energy, DTU) and international, associated with several smart grid labs in Europe. During the first workshop key drivers for software choice to support lab activities were identified, considering the range from research, education to commercial the lab use.

Some insights from the first workshop:

- there are very diverse requirements, both for these different lab users, but also the SG lab use cases vary significantly. Even for similar requirements, very diverse solutions are in use
- a key factor for distinguishing software requirements is the trade-off between “training time” vs. “project time”
  - in *education*, solutions have to be robust and easy-to-use as students typically have little time to learn tools, and technical support is necessarily limited
  - in *commercial use*, ‘robustness’ and ‘short deployment time’ are similar requirements, in a commercial setting, the reduced time can also be achieved by dedicated staff
  - in *research*, ‘flexibility’ or ‘versatility’ is a key factor; here simplicity and robustness are traded off against a necessarily higher level of expertise on the researcher side.
- commercial software and (non-commercial) open-source software seem equally common, though commercial software typically offers “simplicity” and “robustness” rather than “flexibility”.

Following, the domain study [1] structured these and other criteria as a foundation for the lab survey [2], which then attempts to answer two questions: “what is a smart grid lab?” by identifying common focus areas, and “what software is used and needed in such a lab?”. All labs have an individual history with different backgrounds and aims. Yet, based on anecdotal evidence from research overlaps and technical features of 8 investigated labs, three lab stereotypes have been identified:

- Electric & Electronics Lab
- Energy System integration & flexibility Lab
- (Real-time) Simulation Lab

PowerLabDK combines features from all three stereotypes, but also here the separation is still apparent in the current activities. For software aspects, the picture is more complex, and can hardly be summarized: both commercial vs. self-developed solutions are common, interfacing and maturity is at different levels and different use cases are common. Still it is clear that a specific research focus motivates specific requirements for software flexibility at different levels. This need for flexibility is tied to a common picture: all participating labs aim for a high level of software competence among their researcher staff, which seems essential for lab operation.

Workshop 2, which was organized as a mini-conference together with active involvement of workshop participants, was focused on solutions and concrete development steps. Here different technologies (e.g. real-time co-simulation, loose coupling design & simulation) were mapped out against the benefits created for the development and deployment phases. WS2 results are reported in detail in D4.2 [7].

1.2 Survey of Lab Users

An internal survey was conducted in November 2013 among staff of the Center of Electric Power and Energy (CEE), which constitutes the majority of the research and technical staff associated with the PowerLabDK labs. With ca. 40 replies, about 50% of the eligible staff responded to the survey; results of the survey are reported in [4].

The survey results offer greater transparency on the active research practice and associated software use and competencies at CEE. In depth analysis of the results led to several ideas for

potential improvements and initiatives, which are reported in detail in [4]. Here we shall just name a few headlines:

- Research Fields & Common Interests among the five research groups:
    - Three research topics including *optimization*, *simulation technology & algorithms*, and *EV technology* are of interest to nearly every group.
    - Six topics of interest are shared by three of five research groups each.

To better support collaboration and exploit overlaps and synergy, the following aspects should be addressed:

- *Knowledge sharing, information & model exchange*: the most common types of models are power system & components and thermal, economic & statistical models.
- *Data:* a) Size of data sets b) "common" data types: grid data; production, demand & weather data; prices; forecasts; EV charging patterns (user behaviour) & charging spot (GIS) data.
- *Human to Machine - lab operations*: Only a fraction of CEE staff is practically involved in lab work. “on a regular basis” spends less than 50% of their time in the lab; more clearly identifying and assigning responsibility to people more closely involved with lab operations.
- *Teaching*: Lab exercises are part 50% of courses; teaching activities in the lab are asymmetrical across groups; “packaged” software setups could facilitate lab-related student activities; for example, are ‘standard’ configurations of the lab meaningful to facilitate early stage student projects? Removing barriers for student projects, such as at MSc and BSc theses. becomes important to anchor the lab in teaching activities.
- Requirements for *support software*?
    - academic lab users require very different types of experiments and setups for their research
    - most academic users spend only a small fraction of their time in the lab (‘one-time users’),
    - the load on ‘go-to’ persons, who are both researchers and technical staff is rather high to support these one-time users.

Here, in the first place, knowledge-sharing approaches will be helpful. The organization of topical workshops and internal wiki sites could improve information sharing. Further, lab software that balances the following requirements is desirable:
a. supports an API in a programming language which “one-time users” are familiar with
b. flexible and adaptable to a large variety of setups (interfaces & configurations), and
c. facilitates lab configuration and deployment of controllers and software

Repeated types of experiments occur in context of courses and other teaching activities. Here more standardized software setups and lab configurations could also relieve teachers, lab technicians and improve the learning & research outcomes.

1.3 Lab Software Use Cases

The work on use cases has been fruitful to put initial software development ideas in a common framework and in context of lab use. The result is meant to facilitate future lab software improvements by helping communicate ideas in context to find their place in a lab, as well as to communicate development ideas to external stakeholders.

Two levels use cases have been formulated:

1. Lab Business Process (LBP) – as high-level use cases covering most development ideas
2. Software Use Case (SUC) – detailed use cases for a subset of the ideas.

The LBPs address:

1. Development & test of controllers in the lab (6 LBPs)
2. Managing the Lab, Information and Lab software (3 LBPs)

And the SUCs:

1. Co-simulation and development support infrastructure (2 SUCs)
2. Control Software Deployment and Communication Interfaces (5 UCs)
3. Configuration Management (3 UCs)
4. Lab Information Management

The use cases report D2.1 [3] defines key ideas, relevant for further development as well as background for interpretation of the feasibility study and survey results. Some concepts further defined in this report are the *LabOS* and *LabIS,* and *Control Software (CS),* as well as the concept of co-simulation as an 'emulated lab'. A lifecycle perspective on control software development in the lab is introduced, which provides a framework for the results from WS2 (D4.2 [7]), as well as a concept for evaluating the benefits of enhanced lab software support or co-simulation environments (see also Section 2.2).

1.4 Feasibility Studies

Exploring new options and expanding on strengths of PowerLabDK (PLDK) has been a main theme for the feasibility studies:

- *FS1:* Establishing the *system-testing capability of the PLDK Electric Lab* in combination with Intelligent control lab (ABB Network Manager SCADA & Blade Center)
  - remote software deployment and testing; quick on-site preparation
  - several 'hacks' via SYSLAB instrumentation to interface with local components and implement additional measurement
  - SYSLAB software & support facilitated interfacing with several lab DER
- *FS2 & 3:* Extending in-house software with *co-simulation capabilities*
  - follow-up training event on use of co-simulation via *mosaik* (Oct. 2014);
  - initial support for FMI standard for co-simulation.
- *FS4 & 5*: Identifying bottlenecks and potentials for *distributed control systems deployment*
  - For effective deployment of a distributed controller, the development environment should require "distributed system" behavior.
- *FS6* through *9:* Exploring several *new interfacing options* for PowerLabDK:
  - *OPC-UA* (up to functional testing) (FS6)
  - Service-based interfaces (based on *SoA-ML*) (FS7)
  - *OpenADR* (initial development; FS8); now followed up with an innovation activity to develop a simplified API and implementation to demonstrate in a European context
  - Enabling *off-site remote control* via a simple white-board server (FS9)

Further, by recording the time spent on parts of these activities an experience-base is available to estimate future development resources. The feasibility study summary D3 [5] provides a compact overview of alternative development paths and required resources.

2. Synthesis of Outcomes

The above summary focused on the main work streams of the project. In this chapter, we will instead focus on a few key ideas that emerged with the RTLabOS project, but are spread across work streams and reports. These three ideas are:

1. A taxonomy of smart grid labs as ecosystems, instead of just a collections of connected devices and programs
2. To view a smart grid lab in the perspective of the life cycle of control software
3. Advancement of PowerlabDK by specific results and foundations for internal strategy development

The following three sections discuss each of these ideas in light of our findings.

2.1 Taxonomy for Smart Grid Labs: Focus Area, Activities, Competences, then Software

Taxonomy is classification of a topical area so that it becomes more accessible, for example to research or to explanation. This makes complete sense for things of the past, but is harder for living and dynamic things. Biologists do it anyway.

Smart grid labs are a rather new area with a lot of development due to ongoing research and investments. The evolving smart grid lab is therefore a result of past investments, new ideas that propagate into investments, as well as research challenges and skills of lab staff that bring about new developments. As discussed in Deliverable D1.2 (Lab Survey) [2], it is therefore important to formulate a bigger frame to characterize (and eventually classify) a smart grid lab. The process of identifying these features was started with RTLabOS workshop 1 [7], developed into the Domain Study (D1.1) [1], and then applied and evaluated in the Lab Survey D1.2 [2]. The features combined for our analysis have been

- Lab equipment
- Activity types prioritized in the lab
- Research focus area
- Software competence of staff

While the specific software in use is quite diverse across labs, several indicators we developed allow further characterization with respect to software automation of routine lab tasks: *data handling, experiment booking* (and associated security)*, fluency between simulated and physical lab environment* (for controller software)*, hardware-in-the-loop capability,* and *co-simulation capability*.

Further, we defined structured concepts for the actual software systems used inside a lab, such as for lab management (*LabOS*), information sharing (*LabIS*) and control software (*CS*). Their functions have been further developed into generic high-level use cases (we call them *lab business processes*), as well as more detailed use cases (called *software use cases*). The work of formulating these use cases was surprisingly complex, and we believe the resulting report, D2.1 Use Case [3], provides a unique collection of software requirements for facilitating system testing in a smart grid lab.

In this line of thinking, we may summarize that the first main outcome of RTLabOS is a *conceptual map of smart grid labs*, which may provide guidance for strategic development as well as further exploration.

2.2 Smart grid lab for maturing control software

Is there an overall purpose or use case that may provide a metric for the effectiveness for smart grid labs that perform system testing? One major difference between more classical electric power labs and the smart grid labs investigated in this study is a focus on testing software systems associated with control. Control software is any smart grid software associated with automation purposes which can be matured via lab testing, including controls (local and remote, at asset level, aggregation, or SCADA level), protocols, assessment algorithms, online operator decision support and visualizations.

The view of the lab as an environment for developing, maturing and deploying control software is reflected in several use cases and motivated several of the feasibility studies, e.g. on co-simulation. This concept of maturing control software introduces a lifecycle perspective: at the start of this process, a controller may be plainly a concept or prototype (e.g. conceived in simulation environment), and at the end, in field deployment, this concept is embedded with often several layers of software systems. The key parameter to observe in this perspective is the technical risk of deployment: lab-tested software is more certain to function as intended then a mere concept.

For complex software systems, however, rather than moving directly from a concept to deploying it in the field, several iterative development steps are required. Acceleration and facilitation of such iterations must be a key criterion for the effectiveness of lab support software.

This view of the lab as a nurturing, maturing and validation environment inspired fruitful discussions as RTLabOS Workshop 2, and we invite the reader to explore the recorded results in Appendix B of D4.2 [7]. A model characterizing the lab-related stages of control software maturity was used and refined at this workshop. It defines five stages: *A. Concept, B. Development, C. Lab testing, D. Demonstration, E. Field deployment* [3], which each relate to a different role of simulators and the lab as experimentation, testing and demonstration environments.

At each stage the software achieves a higher level of maturity and the technical risk is thereby decreased. In the development stage, a software-based testing environment is much preferred to a lab environment, as rapid iterations of tests are needed. Mockup software interfaces are already common practice for development and configuration testing. At this stage, also co-simulation may be introduced as a development tool to increase the maturity of control software, as reported by several participants in WS2 [7]. If some fluency between (co-) simulation and lab environment can be achieved, also a ‘virtual lab’ based on a software model of the lab can be a strong facilitator for increasing software quality before lab-deployment. Such features can significantly reduce the time needed to spend in the actual lab (as reported in FS1 and FS5; the effect of skipping a stage is reported in FS4 [5]).

These ideas explain why co-simulation studies are equally prominent as lab software developments and lab deployments among Feasibility Studies reported in D3 [5].

The structured reporting of the FS studies, in which a number of development and deployment techniques have been investigated, allows to establish “reference cases” for the time structure of deployment, development & interfacing tasks.

In practice, there is an obvious step when transferring control software from a simulation environment to the lab. However, it is harder to formulate the criteria for maturity that are actually be fulfilled by 'completing' one stage. For the maturity stages outlined above to function as an actual life cycle management tool, such testing criteria would have to be established. It is an important future work for the community of smart grid labs to establish validation requirements for system testing. In other words to answer the question of one Workshop 2 participant: *"How do you validate a control architecture?"*

In this line, we may claim that RTLabOS Phase I contributed to outlining a maturity model and toolbox for control software development, it pointed to potential accelerators as well as challenges to be anticipated.

2.3 Advancing PowerLabDK with new experience, orientation, and international anchoring

The RTLabOS work has contributed to PowerLabDK development in several ways.

Firstly, by practical advancement through feasibility studies. The demonstration of Spirae's BlueFin® established a system deployment capability of PowerLabDK as well as ways of remotely deploying and testing software. It also challenged and matured the OPC connectivity features of the existing ABB SCADA installation, and it confirmed the feasibility of this commercial use case on platform / control software demonstration. The other feasibility studies each established new interfacing and deployment capabilities, and importantly also developed connections to international research units further through co-funded external visits of CEE staff (AIT, Austria; Lawrence Berkeley National lab (LBNL), California; OFFIS, Oldenburg).

Secondly, due to active involvement of several internal user groups in the three RTLabOS workshops [7], as well as feasibility studies and, generated interest in lab-related challenges. A user survey [4] provides insights about CEE staff research activities in association with software use and the lab, providing insights for PowerLabDK coordination and the development of focus groups. Recommendations are targeted at the near-term, pointing toward opportunities for targeted information-sharing initiatives (RTLabOS D2.2 [4]).

Thirdly, with workshops identifying state-of-the-art questions on lab software and related research goals on system testing, RTLabOS contributed to agenda-setting in the context of European smart grid labs.

Finally, the use cases provide tangible ideas that help guiding a SYSLAB development strategy: in an environment with widely different levels of software skills and platform development interest, an internal release of the use cases facilitated discussions on development of support functions. It also opened a perspective for further application of the use case methodology for communication between ICT and non-ICT researchers and technical staff.

Overall, RTLabOS lead to more clarity and transparency on PLDK user needs, concrete experiences with concrete follow-up involving and PLDK development and innovation activities. A side-effect are a better anchoring in the international community of smart grid labs as well as further forthcoming publications.

3. Reflection and Recommendations

In this chapter we reflect on the project's goals and outcomes, and present recommendations.

3.1 Reflection

The initial intent of the project was to develop a fundamental architecture for an integration platform for PowerLabDK. During the start-up phase it was quickly realized that the requirements inside the different PowerLabDK labs for such a platform were too broad, and that use cases for such integration were not sufficiently clear. Also not all challenges to be addressed were of a plain software nature. Further, the 'state of the art' was a) extremely broad and diverse and b) not easily defined, as laboratories do not typically publish directly about the software in use. In publications, only specific components (such as specific real-time simulators) and setups are typically reported. As a result, the project strategy was adapted to achieve the project objectives with a more flexible, agile, approach. Table 1 relates the original goals with the impediments and how the goal has been addressed in the project.

Table 1 Project objectives with impediments and alternative realization

Objective / assumption	Impediment / realized risk	Realization / Workaround
Formulation of architecture for lab integration platform	no single architecture feasible (as anticipated risk); requirements much more diverse and complex than anticipated	Re-definition of deliverables: formal treatment of pre-architecture steps by emphasis on use cases (D2.1), user requirements (D2.2) and functional analysis (D2.1, D3).
State of the Art report	"art" of SG laboratory software infrastructure not sufficiently homogeneous; limited literature	D1.1 Domain Study D1.2 Lab Survey
One post-doc working full-time	No recruitment feasible → delays due to staffing issue	Work with current staff in parallel tracks; extend project time frame; more senior staff
Integration with real-time simulation of FA-ENDK and SOSPO projects.	Direct coordination with two independent efforts: SOSPO and SCADA/RTDS coupling not feasible	SOSPO concepts modelled as use case (LBP4, D2.1), reviewed by SOSPO team member; involve-ment in workshops; e.g. WS2: SCADA&Operator Support (D4.2)
Focus mostly on internal user groups	Higher interest from international workshop participants	Utilize international competence & interest for analysis and ideation (D1.2; D4.2)
Dissemination: workshops, 3 conf. papers, two articles, public reports	Due to change of report strategy & staff: articles not completed in project time frame; reports prioritized.	Workshops, reports, and conf. papers addressed; article finalization delay accepted. Additional dissemination material: website and videos.
Collaborate with SG labs to avoid 'competitive' angle	No competition on RTLabOS scope realized	Good collaboration with several laboratories established via workshops and feasibility studies.

In reflection on these original objectives, it can be concluded that there is not going be *definite architecture* for a smart grid lab in line with the RTLabOS vision. Instead, RTLabOS Phase I contributed to establishing a network and knowledgebase at PowerLabDK, as well as a strategy for systematic further development along the lines of the RTLabOS vision. Instead of a one-shot effort, further improvements to the PowerLabDK software infrastructure will follow specific needs, but now can be facilitated and strategically guided by the insights reported here.

The outcomes of RTlabOS are reported in this and six further reports:

- D1.1 - The Requirements Domain for Laboratory Software Infrastructure [1]
- D1.2 - State of the Art Smart Grid Laboratories [2]
- D2.1 - Use Cases for Laboratory Software Infrastructure [3]
- D2.2 - User Survey and Characterization of User Profiles and User Requirements [4]
- D3 - RTLabOS Feasibility Studies [5]
- D4.1 - RTLabOS Phase I: Software Infrastructure for Smart Grid Labs *(this report)*
- D4.2 - RTLabOS Dissemination Activities [7]

The reports are kept compact and separate as each report may serve an independent purpose. We suggest that this overview report provides the orientation to look deeper into the results.

3.2 Recommendations

Two sets of recommendations have been identified for further work: The first set suggests further steps at PowerLabDK and CEE; the second set refers ideas for further research and development initiatives.

The first set of recommendations addresses PowerLabDK practices in general:

R1. *Keep staff software competence up – no research platform is 'stable'.*

For research in smart grids many developments are software-based components. With continuous development, software in a smart grid lab cannot be addressed with a fit & forget approach. Any manual, how-to or other documentation will become outdated. Staff IT competences therefore need to be strong and addressed systematically for academic as well as technical staff. In particular, there should be qualified staff dedicated to (software) development, with direct involvement in research to keep up and ensure alignment between research goals and infrastructure. This could for example be implemented by identifying some non-critical but challenging software projects that can largely be handled by 90% developers and 10% lab personnel (academic or otherwise). Making these projects the defaults for the development staff to be working on would allow them to get called to support an experiment/demo/upgrade etc.

Formulation of these projects for continual improvement of the lab can be a process involving non-development staff, facilitated by the Lab Use Cases developed in D2.1 [3].

R2. *An information sharing strategy is necessary, not optional.*

In small labs, information sharing works best directly from person-to-person, in particular if all staff is directly and frequently involved in lab activities. At CEE, the larger fraction of staff is only rarely involved in lab activities. The resulting "go-to persons" end up spending their time helping others, which is unaccounted for in project work. This applies to research staff as well as to technical support. Another problem with this strategy is that all organizational learning remains

with few staff members. Alternative strategies, such as the participative information sharing in topical circles, in a wiki form, or via more systematical meta-information repositories are discussed in [3] and [4].

R3. *Start organizing information on experiments and typing of data and information*

Several advanced use cases (e.g. LBP7 or LBP5, [3]) require well-structured and typed information about the lab environment of an experiment. Also effective use of model-based approaches for interfacing with lab equipment require a structured approach to naming of signals (SoA-ML, also OPC-UA, FS6 & FS7 [5], or IEC61175, as presented in WS2 [7]).

It is challenging to introduce formal conventions in a research environment, especially if rapid development is the norm. Not using conventions, however, has a similar effect as not having an information sharing strategy: bottlenecks are created by every new development as there is no implicit coordination on the basis of the accessible information alone. Conventions need to be introduced as an element of common practice, and cannot be expected to succeed on first attempt.

Key to an effective use of conventions can be interdisciplinary work, such as collaboration between software engineers and power engineers. Collaboration across locations and research focus can be a similar driver, which should be used if it happens anyway. A careful approach to introducing some formality is feasible, systematic naming conventions are powerful and mark the way forward to a more integrated lab.

R4. *More system testing and demonstrations in PowerLabDK Labs in Lyngby*

FS1 [5] clearly proved the capabilities of the lab, but also that the know-how for developing such a setup was available in SYSLAB. Compared to SYSLAB, however, the Electric lab is closer to potential audiences; because it is compact, it allows an audience to more easily grasp the dynamics of an experiment. Further, with the potential of controlling the amplifier, also in closed-loop with the RTDS, quite advanced scenarios can be envisioned. All these features may be employed for advanced system testing and demonstrations. Yet, even with simpler setups, attractive demonstrations and could bring in future customers, colleagues, researchers and students.

R5. *Standardized interfaces are great, but choose carefully which to support.*

At first sight, several IEC 61850 implementations are available at CEE; on paper, ABB's network manager supported OPC-DA; and since RTLabOS, PowerLabDK also supports web services via SoA-ML (partly), OpenADR, and OPC-UA (both under development).

However, after a closer look at the evidence, many of those standards are only supported in part. Modern industry standards are complex, and fully supporting a standard means a continuous development to stay compliant as the standard evolves. In practice for research software, it is much easier to support and maintain proprietary lab interfaces and low-level established standards, also for deploying external software (as long as developers are involved on both ends); FS5 made a case here; FS1 made a case for the simpler/lower-level interface (Modbus). Adaptability and low complexity have been key in such cases.

Fully implementing a modern standard makes sense only if there are significant use cases, such as for testing with commercial "black box" equipment. While at SYSLAB that has not applied so far, the alternative for a research lab is to support a modern standard as early-adopter, to identify weaknesses and limitation and thus to contribute to the standard's evolution.

It might be worth focusing on some standards in the smart grid domain, but understanding your "customers" and research purpose helps picking the right ones.

R6. *Co-simulation is a powerful development tool, but don't start duplicating all interfaces.*

Several smart grid labs already employ co-simulation as a research and development tool. However, there is no silver bullet: for development it is more important to be practical than sophisticated. Co-simulation as a development tool requires equipping both control software and simulators with interfaces adapted to the orchestrator. Whereas loose coupling approaches are more straightforward to handle at the expense of being less scalable, sophisticated co-simulation may further require a special formulation of controllers (FS3, [5]). As SYSLAB supports built-in simulated behaviors (e.g. FlexHouse simulator) and loose coupling (e.g. with mockup SYSLAB nodes; FS1, FS5), developing a wrapper for including (mockup) SYSLAB nodes into a co-simulation may be more effective for development purposes than developing dedicated simulation models for SYSLAB assets. With nodes in simulation-mode, co-simulation wrappers could then allow network domains (electricity, heat, communication) to be integrated via simulators. From our experience (FS2 and follow-up), mosaik has been a powerful and sufficiently easy to use tool for such a purpose.

In this way, the vision of a 'virtual lab' could be realized incrementally by developing simulation models of lab network domains, alongside further improved 'simulation-modes' for SYSLAB nodes. While this approach suits both the use cases of development support and 'virtual scaling of experiments' [3], it is primarily suited for real-time approaches. As noted above, a fully embedded co-simulation requires architectural modifications to the simulated entities.

<u>Recommendations to target further research and development initiatives:</u>

R7. *Develop metrics and processes for control software maturity and connect to industry.*

Development and testing is not a formal process in most research contexts, but it is necessarily one for industry. However, for complex control software (e.g. distributed resource management; aggregator software; control & decision support) there is no standard process. With increasing maturity of the smart grid domain, smart grid labs can have a role in facilitating the deployment for demonstration, but also for testing of such software. Further, infrastructure for scalability and cybersecurity tests are relevant. To support such developments, the idea of control software maturity requires further development on assessment frameworks, metrics and indicators as well as testing procedures.

R8. *Follow up on system testing.*

RTLabOS contributions centered on interfaces and platform support for system-testing as well as processes surrounding and control software maturity. Validation and verification of test requirements and standards has not been in focus. More rigorous definitions of lab infrastructure and test requirements need to be formulated for a rigorous testing framework. DERLab e.V. and ISGAN are two networks in the smart grid context in which such initiatives have been launched to support increased system-testing in labs.

R9. *Statistical survey and database on smart grid lab capabilities, not just inventory*

Current databases and surveys on smart grid labs are very infrastructure-oriented, with a focus on an inventory of components and supported standards. Considering the RTLabOS experience on the variety of interpretations of standards (see also R5), and the many other factors found in

the Lab Survey D1.2, inventory-based lab descriptions are insufficient for evaluating and identifying appropriate testing and demonstration labs and facilities. Using the new indicators developed in D1.2, a new survey could be formulated that is more focused on the actual capabilities of the labs. With a streamlined questionnaire a larger number of smart grid labs could surveyed statistically, both to measure the state of the art and developing progress indicators. Wuch a survey should be supported by international lab networks such as those mentioned in R8.

R10. *Research "controller container" for facilitated development, lab testing, field deployment.*

For embedded systems and especially embedded controllers, development of controllers can be performed in a high-level language on a PC-based simulation platform and then be deployed to a 'real-time target' by compiling the controller to machine language onto a DSP chip. Such solutions (e.g. LabView® or dSPACE®) are common and have also been reported in our workshops (see D4.2) and the lab survey (D1.2). Even a hierarchical distributed controller can be automatically deployed on custom platforms, as demonstrated by Spirae's Bluefin® platform in FS1 (see D3). Code generation and library components can also developed for the function-block standard (IEC61499, see D4.2). Such 'model-based' and 'code generation' techniques are powerful facilitators, but they are limited in the complexity of control software that can be handled effectively. Further paradigms for controller migration, such as the loose coupling via a generic message bus (SMB, WS2, see D4.2) offer more flexibility on the coupling but less support infrastructure. More formal approaches via domain specific languages (DSLs) may be employed to further facilitate and simplify specific sub-tasks of control software development. Smart grid control software, in the definition employed in RTLabOS, encompasses a wide range of requirements (e.g. market-based distributed control) which exceeds the cases mentioned above. For example, enhancements could be required to enable co-simulation of distributed controllers with communication systems. However, seeing the advantages of facilitated development approaches, there is significant potential for accelerated development and testing. We see a potential for further valuable research into this field.

References

[1] A. M. Kosek and K. Heussen, "D1.1 - The Requirements Domain for Laboratory Software Infrastructure," Department of Electrical Engineering, DTU, Kongens Lyngby, 2013.

[2] K. Heussen and O. Gehrke, "D1.2 - Lab Survey “State of the Art Smart Grid Laboratories," Department of Electrical Engineering, DTU, Kongens Lyngby, 2014.

[3] K. Heussen, A. Thavlov and A. Kosek, "D2.1 - Use Cases for Laboratory Software Infrastructure - Outline of Smart Grid Lab Software Requirements," Department of Electrical Engineering, DTU, Kongens Lyngby, 2014.

[4] J. Hu and K. Heussen, "D2.2 - User Survey and Characterization of User Profiles and User Requirements," Department of Electrical Engineering, DTU, Kongens Lyngby, 2014.

[5] K. Heussen, A. Thavlov and A. M. Kosek, "D3 - RTLabOS Feasibility Studies," Department of Electrical Engineering, DTU, Kongens Lyngby, 2014.

[6] K. Heussen, "D4.1 - RTLabOS Phase I: Software Infrastructure for Smart Grid Labs," Department of Electrical Engineering, Kongens Lyngby, 2014.

[7] A. M. Kosek and K. Heussen, "D4.2 - RTLabOS Dissemination Activities," Department of Electrical Engineering, DTU, Kongens Lyngby, 2014.