ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

CERN CONTROLS CONFIGURATION SERVICE – A CHALLENGE IN USABILITY

L. Burdzanowski†, A. Asko, A. Lameiro Fernandes, K. Penar, C. Roderick,
B. Urbaniec, V. Ioulios Vasiloudis, CERN, Geneva, Switzerland

Abstract

Complex control systems often require complex tools to facilitate daily operations in a way that assures the highest possible availability. Such a situation poses an engineering challenge, for which system complexity needs to be tamed in a way that everyday use becomes intuitive and efficient.

The sensation of comfort and ease of use are matters of ergonomics and usability - very relevant not only to everyday equipment but especially software applications, products and graphical user interfaces.

The Controls Configuration Service (CCS) is a key component in CERN's data driven accelerator Control System. Based around a central database, the service provides a range of user interfaces enabling configuration of all different aspects of controls for CERN's accelerator complex.

This paper describes the on-going renovation of the service with a focus on the evolution of the provided user interfaces, design choices and architectural decisions paving the way towards a single configuration platform for CERN's control systems in the near future.

INTRODUCTION

For a long time, the subject of ergonomics and usability in the software domain was mainly attributed to consumer products. Business products and industry-oriented software in particular received less attention to such aspects, often being considered as secondary or optional with respect to the core functional needs. This situation is gradually changing due to gaining a better understanding of the importance of usability as well as advancements in modern software technologies aimed at easy and efficient development of user-friendly interfaces. In general, Control Systems used in particle accelerators and large experiments can be perceived as industrial installations where stability, efficiency and reliability are primary concerns (including user-interfaces and software applications). However, in the case of scientific experiments and installations there is no hiding the fact that the primary mission is not monetary profit but rather the pursuit of greater ideas and fundamental understanding. This mission is often conducted by personnel for whom the task to control complex installations via software application is performed in addition to their principle scientific or engineering background. This aspect should be seen as a motivator for Control System designers and engineers to provide not only reliable and efficient solutions, but also ergonomic and usable software applications to facilitate their usage as the primary tools for daily work.

† Lukasz.Burdzanowski@cern.ch

The CERN accelerator Control System has evolved together with the accelerator complex. Certain software applications and user interfaces have roots going back to the early 1990's. This situation naturally poses a challenge with respect to maintenance and further evolution of the software. The first basic graphical user interfaces provided by the Controls Configuration Service (CCS) were developed in the mid–1990's and have evolved ever since [1]. The specificity of the CCS service in the CERN accelerator Control System is its centralised role making it a hub of configuration data among all Control System layers. Having this role, together with a broad scope spanning many domains, with a large number of users (~500) – creates a major challenge in providing and maintaining usable, user-friendly interfaces without imposing unnecessary constraints on the user community.

The challenges to design and provide ergonomic and user-friendly Controls applications, practical strategies and design concepts and an overall discussion of usability aspects with respect to centralised core Control systems services like the CCS are discussed further in this paper. The practical experience gained during development of the new generation of CCS tools serves to reflect on how Controls applications can be designed and developed such that long-term maintenance and user satisfaction are not conflicting goals. In addition, technical highlights about deliberately selected software technologies bring insights on how graphical user interfaces can be rapidly developed without sacrificing responsiveness or ergonomics.

USABILITY

Generally speaking, the term *usability* expresses facility of use or ease of learning for a given man-made object; let it be a tool or a device [2]. In a world of software engineering the term usability represents the degree to which a piece of software can be used by its end-users in a satisfactory, effective and efficient manner. A given user-interface can be considered intuitive when it does not require intense training and exhibits a natural workflow for which it was designed. The subject of usability is widely acknowledged in the software industry world and is formalised by dedicated ISO standards: ISO/TR 16982:2002 and ISO 9241-210:2010.

Software applications, specifically graphical user interfaces (GUI) are by no means mere tools facilitating daily work. Their sole purpose is to establish an interface between human users and "machines", specifically the internal implementation of a system. This point is especially important – one of the common failures of some GUIs is an inherent lack of abstraction between system internals, (e.g. a database structure) and its

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

representation to the end-user. While occasionally such a case may be justified, in general end-users don't want to be exposed to plain structures of tables, rows, columns nor any other implementation details. Rather what is expected is that a software application and its GUI(s) expose real practical use-cases and every-day workflows [3]. As an example, following figure presents a screenshot of Controls Configuration Data Editor (CCDE) – an application discussed in more details through this paper (see Fig. 1).

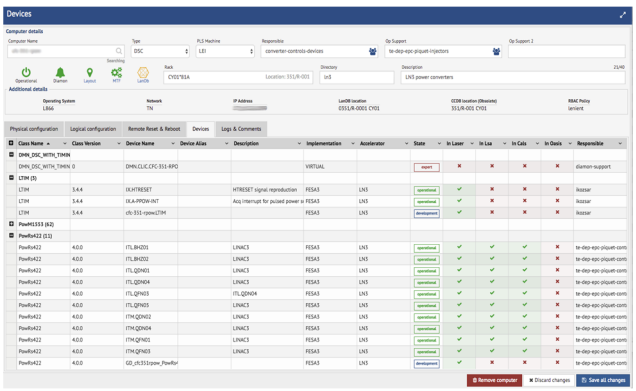

Figure 1: CCDE view of control front-end software devices.

What we can identify here is a clear separation between root context in which user is working: Front-End Computer in this case, with its details like responsible, role, type and below it more specific information, accessible when necessary but not immediately shown.

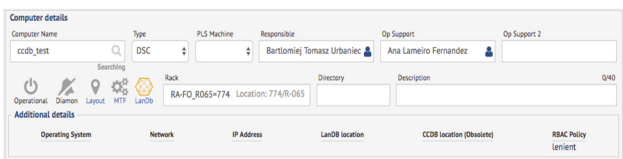

Figure 2: Focus on a root context – a front-end computer.

The root context (see Fig. 2) stays fixed while the user can easily change between different details: physical configuration of a FECs (hardware equipment), logical (startup processes), hosted devices and even history of changes and comments of control system experts. Context of hosted devices is especially important since it presents not only devices themselves but their state and relation to other components of the CERN Control System, specifically to high-level control system services (see Fig.3).

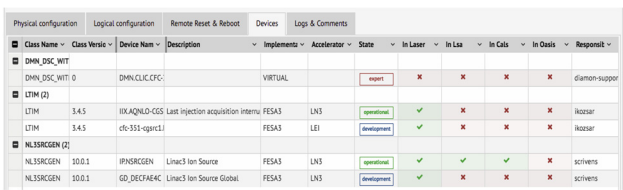

Figure 3: Clear separation of information dependant on a root context – a summary of software devices hosted.

Clear and well-defined structure of the screen maintained across the application, easy access to detailed information without loss of the core context, built-in integration with various component as well as intuitive colours and symbols – all these aspects contribute to increased usability.

An application is a tool that is expected to facilitate work while not generating unnecessary or artificial steps to accomplish common tasks. Therefore, the real challenge to achieve usable and ergonomic applications is to concisely capture user expectations, real-life needs and provide a means to facilitate daily work. This is especially challenging in the case of engineering-oriented systems, for which the primary concerns are efficiency, robustness and minimalism – leaving look-and-feel as a secondary need. This is precisely the challenge faced by the CCS.

Aspects of User-Interface Usability

Visual consistency is one of the most evident aspects of a highly usable system. By streamlining the look and feel of typical graphical components: data grids, panels, buttons, location of elements and general application layout we can greatly lower the learning curve for users, and more importantly reduce effort needed to understand and follow application behaviour. Unified typography, symbolism and use of icons or other pictographs, colour palette – are all equally important and can be perceived as primary utilities in establishing a consistent visual experience. In the example below (see Fig. 4) – colours and icons used in on-screen notifications are streamlined within the application. Separation between an affirmative message, a warning or a system error is evident and adequately captures user attention.

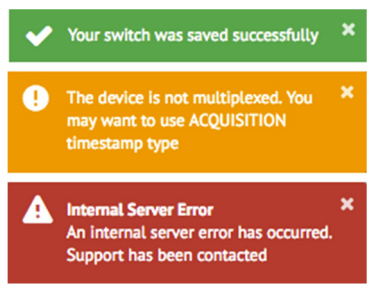

Figure 4: An example of application notifications.

Lack of visual consistency is one of the main factors contributing to user dissatisfaction and strain in their daily work. Moreover, visual inconsistencies or ambiguous behaviours are often not identified directly but stand out more as a general negative "feeling" or perception for the end-users.

One particular aspect of visual consistency is the notion of concise behaviour of the application interface. Next to the streamlined presentation of UI components underlying actions need to follow unified rules and schemes. The flow of actions should be logical and aligned with natural work scenarios while still being able to hide underlying technical implementation aspects from end-users. Actions in the system should be limited in steps and closely

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

connected to the context in which a user is operating. As an example, options of single edition of a system entity should be aligned with related bulk operations. In both cases users should be prompted with similar dialogues or warnings. The notion of warnings is especially crucial as a highly usable application should not allow critical errors, for example by letting users inadvertently change critical system data without realising.

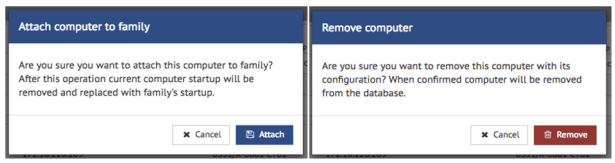

Figure 5: CCDE Confirmation dialogues prompting user.

Fig. 5 shows an example of how visual consistency and concise behaviour helps. Confirmative and regular action is represented across the application always with the same colour and icon. Potentially dangerous action of deletion is explicitly marked in red. This scheme is kept across the application and actions of high importance are highlighted attracting user attention and when justified provoking additional steps (e.g. confirmation). In addition, the dialogues provide auxiliary details and let re-consider an action.

The granularity of a user interface together with the expertise level necessary to operate it effectively are further usability factors. Too coarse-grained interfaces (e.g. panels with numerous inputs of varying data types and importance), tend to provoke mistakes and contribute to mental tiredness. Too fine-grained interfaces or actions on the other hand provoke annoyance and stress, especially for expert users. Logical separation of system tasks should be clearly reflected in the user interface but not in a way where abstraction of the interface is lost. Technical details of the underlying services and programmable applications interfaces (API) should not steer design of the presentation layer in the system. This aspect is essential when evolving and re-factoring existing applications due to functional or technical reasons. Such scenarios are opportunities to revise and scrutinise general design and application flow based on real-life experiences from the users.

CHALLENGES OF USABILTY

Control systems naturally evolve alongside the progressing development of accelerators and software technology. We can observe a trend in which user-oriented applications become high-level tools, increasingly visual and build around workflows helping to guide the users.

Users are progressively being given a role of an observer and supervisor of the system while low-level actions are automatized and abstracted. This abstraction brings software engineering products closer to the domain of Control Systems tailored to specific cases like accelerators physics. The domain is not static but its rapid development does not always imply a need for having an equally dynamic evolution of high-level controls system applications. Foremost, the applications need to enable less downtime and more throughput in terms of the amount of work possible to do in a finite amount of time. Highly usable applications are a prerogative in this area.

The diversified and complex nature of Control Systems is one of the challenges to overcome. The natural complexity of CERN's accelerator domain often prevents off-the-shelf products to be easily integrated or customized to meet user expectations. Complex systems require a thorough understanding of the underlying domain in order to properly capture needs and provide software solutions to help in daily tasks.

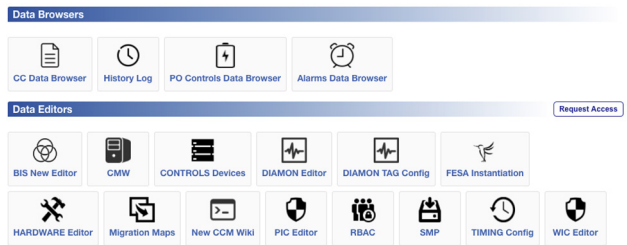

Figure 6: Example of poorly structured applications exposing users to system complexity.

Above figure shows a welcome screen of formerly provided CCS tools. Immediately we can notice that as users we are exposed to a myriad of various applications which indicates how complex and diversified is the underlying system (see Fig. 6). Variety and diversification are places where application providers and designers can gain most when helping users. Properly structured interfaces with well-integrated authorization schemes can guide users through the system and actually help in hiding the complexity.

Development of Control Systems and applications is practically tied to the operational schedule of accelerators and experiments. Any major changes need to follow the schedule thus imposing an evolutionary development approach. As the evolution may be prolonged in terms of time, technical skills and expertise necessary to progressively advance, the applications may become incompatible with market needs and skills available on the market. Control Systems have life-spans stretching from a few years up to 15-25 years of evolution, i.e.: when looking at lead-time between the first design concepts of an accelerator and its full operational capacity or accomplishment of a planned physics run. Such systems are never fully finished nor completed following natural development and advancement of technology and science. With the high inertia of Control Systems, software development and use of technical solutions and cutting-edge market products should be considered with caution since the life-span of software frameworks and specific technologies is noticeably shorter than Control Systems themselves. This situation makes it necessary to evaluate a trade-off between the choices of technologies that are relevant to the skills on the market while not posing a risk of early obsolescence.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

Technological advancement within the software industry can indirectly pose a challenge although not directly related to applications usability but rather development of software interfaces in general. Alongside rapid expansion of the Internet, the Web browsers, development frameworks and libraries targeted at Web applications keep expanding and maturing – providing more facilities to easily develop interfaces which are responsive and provide a rich user experience thanks to advanced UI controls. While not directly related to the usability but rather software engineering in general, providers and maintainers of Control System applications need to carefully evaluate the lifespan of a given software package versus its expected utilization even before starting development. Control System interfaces should not lag behind commonly established UX standards to which users get accustomed to on a daily basis, e.g. consumer products such as e-banking and travel booking applications. Software libraries providing / facilitating use of modern UX trends have to be selected with care as the lifecycle of Control Systems are usually far longer than those of many potentially interesting software choices. Trends and standards may change or evolve faster than the software applications trying to embrace them in order to improve the user experience. This is one more challenge in providing highly usable applications, where a balance between maintainability of software and a dissonance between work-specific and daily life products is not unnecessarily enlarged.

DESIGN STRATEGIES

Wire-framing or mock-ups are one of the most efficient tools to help increase usability. Already at the early stages of analysis and preparation, potential problems can be identified or compliance of a given user interface with globally established rules can be validated. Mock-ups are an indispensable utility in discussions with target end-users. Considering their relatively low development cost, mock-ups should be seen as a standard step in the process of designing applications where usability is at stake.

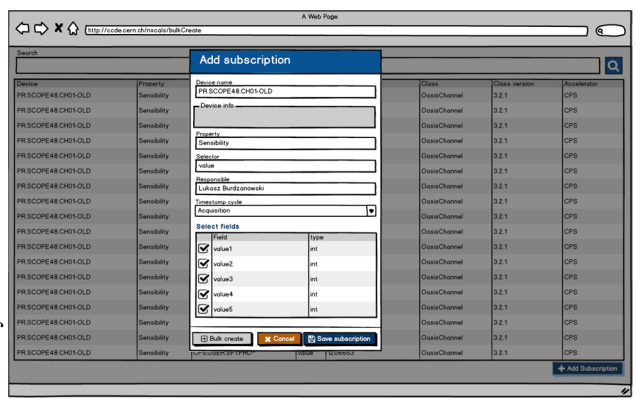

Figure 7: Mock-up of a CCDE view. Please note how user context is focused on data edition by using a modal window with an overlay. Once again buttons, colours and icons remain consistent.

Mock-ups help to deliver applications matching the needs but are not sufficient to assure that shipped products are fully covering the needs (see Fig. 7). The main product life-cycle phases: design, development and maintenance should take into account that lack of usability in some areas is fully justified knowing that the given part will evolve based on changing requirements or emerging needs. It is often more effective to ship early and deal with consequences later provided that an application does not lack in major usability and functional aspects. This case is crucial for new products addressed at large communities.

Managing expert-level access or the multitude of roles that advanced users need to have in the case of Control Systems poses a particular challenge. Such users normally need access to a wide range of operations and functions of the system based on the situation in a given moment. Taking this into account, the provided applications need to support both task and role oriented models. Such applications and corresponding system design ensure that the usability, functionality and maintainability of the applications (and the system itself) remain at a satisfactory level for all kinds of users, and also stay stable across different contexts of work. This concept can practically be realized via well-integrated application authorization schemes that naturally aim to limit access or functionality based on roles. Moreover, this can be based on not only simple editor/expert roles but also on system specific profiles i.e.: middleware expert, front-end software editor, hardware expert, etc. (see Fig. 8).

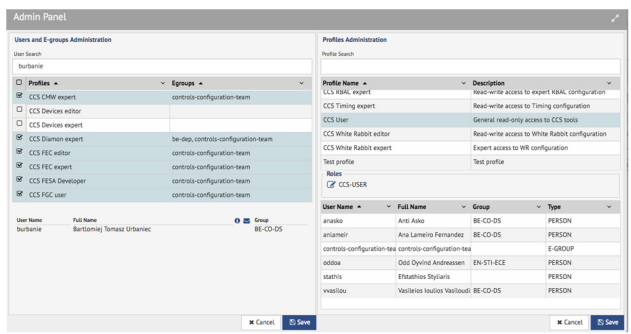

Figure 8: Admin panel of CCDE showing application profiles of the selected user and profiles management.

Having profiles of roles can help application software developers to better structure the application, whilst Control System experts and designers can properly capture real roles and function of the users at runtime. This approach can noticeably lower the mismatch between the application design and practical needs. Expert users having a multitude of roles (falling into multiple profiles) can benefit from built-in user-interface perspectives that let the user change current "profile" depending on their current needs. This usability concept is well established – for example in CAD/CAM tools or software development IDEs – enabling developers and designers to change context and the focus of the application based on interim needs.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

CERN CONTROLS CONFIGURATION SERVICE

The CERN Control System is a data-driven multi-layer infrastructure [4] including:

- *Low-level hardware and software* – e.g. timing infrastructure, equipment drivers, Front-End Computers (FEC), end-user developed C/C++ binaries representing operational "devices", etc.
- *Middleware layer* – e.g. read/write access to processes running on FECs and Role Based Access Control (RBAC).
- *High-level software* – e.g. high-level settings management, data acquisition and archiving.

The Controls Configuration Service (CCS) helps bind all of the layers together by providing them with complete and coherent configurations that are necessary for the proper functioning of the Control system.

The current architecture of the CCS is based on:

- An Oracle database (2-node RAC cluster)
- A set of high-level client Java APIs
- Database level client APIs (PL/SQL interfaces)
- Controls Configuration Data Editor – a new generation unified high-level graphical user interface facilitating data management.
- Numerous legacy GUIs based on proprietary technology (Oracle Application Express – APEX), which are being phased out.

The CCS exists since 35 years, during which time the scope, architecture, technology and development methodology have kept evolving. In mid-2014 the first major service-wide renovation and overhaul started – marking the beginning of a new chapter in the CCS's long history [5]. One of the main goals of this on-going consolidation is a complete overhaul of all related GUIs, including read-only data browsing tools and APEX based data edition tools. Historically, the CCS and underlying components of the Control System have been represented as a set of separate applications and components which did not capture typical workflows, frequently exposing internal details like database table structures, having confusing GUI controls and presenting inadequate error messages. In order to address these flaws and to improve usability and ergonomics, a new tool – the Controls Configuration Data Editor (CCDE) has been designed from scratch with the goal of providing a single and user-oriented view of all Control System configuration data.

Challenges and Solutions

One of the key challenges of the CCS is its diversified and vast groups of users, functions and a multitude of domains and sub-system-specific extensions. Although such a situation brings a considerable level of complexity the benefits are clear and counter potential problems: by having a single and unified source of the configuration in the Control system we are able to assure its consistency, correctness and remove unnecessary and potentially dangerous redundancies. Thanks to the unification, the users of the CCS can access and rely on the same configuration data independently of their role: hardware installation experts, equipment experts, FESA [6] developers or middleware specialists. Furthermore, high-level components of the Control System can reliably depend on the consistent and correct configuration, i.e. InCA/LSA [7] (accelerator settings management) or CALS [8] (CERN Accelerator Logging Service).

The technical implementation of the Control System where well-defined distinct services and components work together should not be exposed to users – especially with respect to GUIs. As a specific example with unified access to configuration data through a single application (the CCDE), users of systems like CALS are able to configure data logging directly when browsing control devices, their properties and fields.

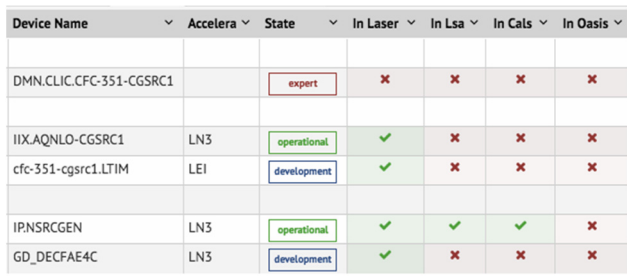

Figure 9: Focus on devices state and role in the system.

The CCS captures not only the definition of a device class and its properties but also how a device is used in a given context, for example as a source of data fed to the Logging system or presence in LSA (see Fig. 9). This is especially valuable when we reflect on the fact that experts responsible for physical equipment (producers of data, e.g. beam instrumentation equipment like wire scanners) are often not the same people interested in specific data during periods of accelerator operation or machine development studies. This is a simple example of increasing usability and system ergonomics for end-users, while at the same time hiding implementation details like integration between separate Controls sub-systems.

A more advanced example is the configuration of Front-End Computers (FEC). Today, in order to prepare a FEC for operational use, several distinct actions are required: put simply – Hardware (HW) experts need to import or declare the networked device in the CCS, configure its physical components like crates and modules

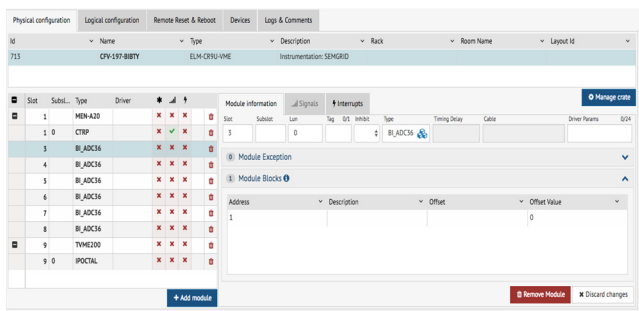

Figure 10: Front-End Computer physical configuration.

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

Above figure (Fig. 10) shows physical configuration with focus on hardware modules locations (slots and sub-slots) as well as summarising configuration of each module by indicating presence of module signals, interrupts, exceptions. Users are not forced to traverse each of the modules to find one which has signals registered – instead all the information is presented at glance.

Next, low-level software experts may then need to setup drivers for HW modules. Finally, equipment experts or FESA developers need to configure FEC start-up sequences in order to launch their FESA processes on the FEC. This task-specific context is presented on following figure (see Fig. 11).

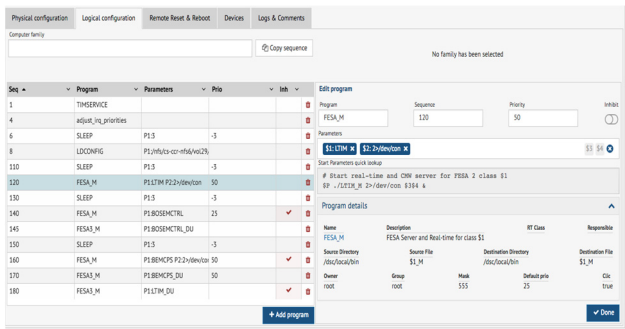

Figure 11: FEC logical configuration - summary of run-time processes configured for deployment.

In the past, all of the above operations required users to switch back-and-forth between different views, tabs and screens of the provided tools, often in a very unintuitive manner. After closer analysis and discussions with the different users taking part in this scenario, two distinct roles were identified in the system: HW installation expert (performing physical configuration) and FEC start-up configurator (responsible for the logical configuration). These roles frequently correspond to different users working in close collaboration.

The CCDE aims to cater for the two aforementioned roles in the way the GUI is designed and works in the context of a currently selected FEC. The feedback gathered proved to be indispensable in this regard and showed clearly that from the perspective of hardware installation which normally involves the setup of a crate (VME, KISS-2U-CRATE, etc.) and its modules (i.e.: a set of CTRVs), the ergonomics of the GUI can be noticeably increased by showing all the relevant information in a single well-structured view despite the large amount of attributes involved. Attention to the logical order of how the information is presented on the screen further increases ergonomics – removing the need for users to switch context and traverse between various tabs and panels. By considering the same FEC and switching the context to the logical configuration users are able to "switch" the perspective on configuration data intuitively and seamlessly.

Technically, the CCDE is implemented as a single-page Web application built on top of modern de-facto standard software industry solutions like Java and Spring (back-end) and AngularJS with TypeScript (HTML5 front-end). Thanks to a stateless architecture, the application is deployed in a high-availability manner, whereby redundant server nodes assure non-interruptive operations for our users despite frequent releases of the new features and improvements. With releases scheduled on a weekly basis, the delay between users submitting feedback and practical implementation is kept to the minimum – increasing users satisfaction and lowering the likelihood of introducing errors in production.

CONCLUSIONS

The question was raised whether usability and ergonomics of GUIs used in Control Systems is a prerogative or should rather be seen as a secondary concern with respect to core application functions. In practical terms the aspect of application usability is equally important as stability or robustness. End-users need reliable and efficient tools that are also easy to learn and use. The CCS examples described show how the challenge of design and development of highly usable applications can be overcome without ignoring core functional aspects of the application.

Above all, Control system applications should capture the natural workflow of the target users in order to help in their daily work. The challenge to provide easy to use interfaces or tools is not only limited to complex domains – in most cases, the challenge is to better understand requirements and real-life scenarios in which future users are going to work. It is a failure of a software engineer when a tool or an application does not fit its purpose. The specificity of CERN's large and complex domain of accelerators and experiments makes it difficult to guess and envision how all CCS users would like to work. Fortunately, software products are generally easier to change and improve than physical entities. It may be hard or impossible to modify already installed equipment but software products give more flexibility in this area. An open-minded software engineer will listen and understand a user's needs. After all, the purpose of software tools is to help, but only users can help software engineers to understand their real needs.

REFERENCES

[1] J. Cuperus, *et al.*, "Integration of a Relational Database in the CERN PS Control System", ICALEPCS'97, Beijing, China, 1997, ID085
[2] D. Norman, The Design of Everyday Things – ISBN-13: 978-0-465-06710-7
[3] S. Krug, Don't Make Me Think - ISBN-13: 978-0321344755
[4] R. Gorbonosov, The Control Systems of the Large Hadron Collider, CERN Academic Training Lecture Regular Program, http://cds.cern.ch/record/1605201
[5] L. Burdzanowski, "The Renovation of the CERN Controls Configuration Service", ICALEPCS'15, Melbourne, Australia, 2015, MOPGF006
[6] M. Arruat *et al.*, "Front-End Software Architecture", ICALEPCS'07, Knoxville, Tennessee, USA, 2007, WOPA04

ISBN: 978-3-95450-193-9 doi:10.18429/JACoW-ICALEPCS2017-TUBPL01

[7] G. Kruk *et al.*, "How to Successfully Renovate a Controls System? - Lessons Learned from the Renovation of the CERN Injectors' Controls Software", ICALEPCS'13, San Francisco, CA, USA, 2013, MOCOBAB05

[8] C. Roderick *et al.*, "The CERN Accelerator Logging Service - 10 Years in Operation: A Look at the Past, Present, and Future", ICALEPCS'13, San Francisco, CA, USA, 2013, TUPPC028