ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV021

UPGRADING THE NATIONAL IGNITION FACILITY'S (NIF) INTEGRATED COMPUTER CONTROL SYSTEM TO SUPPORT OPTICAL THOMPSON SCATTERING (OTS) DIAGNOSTIC

A. Barnes, A. Awwal, L. Beaulac, B. Blackwell, G. Brunton, K. Burns, J. Castro Morales, M. Fedorov, R. Lacuata, R. Leach, D. Mathisen, V. Miller Kamm, S. Muralidhar, V. Pacheu, Y. Pan, S. Patankar, B. P. Patel, M. Paul, R. Rozenshteyn, R. Sanchez, S. Sauter, M. Taranowski, D. Tucker, K. C. Wilhelmsen, B. Wilson, H. Zhang
Lawrence Livermore National Laboratory, Livermore, USA

Abstract

With the ability to deliver 2.1 MJ of 500 TW ultraviolet laser light to a target, the National Ignition Facility (NIF) is the world's most energetic laser. This combination of energy and power allows the study of materials under conditions similar to the center of the sun. On fusion ignition experiments, plasma generated in the interior of the target shell can detrimentally impact the implosion symmetry and the resulting energy output. We are in the final stages of commissioning a significant new diagnostic system that will allow us to better understand the plasma conditions and improve our symmetry control techniques. This Optical Thompson Scattering (OTS) system consists of two major components: a probe laser beamline capable of delivering a world first 1 J of energy at 211 nm, and a diagnostic that both reflects the probe laser into the target and collects the scattered photons. Between these two components, the control system enhancements required integration of over 450 components into the existing automation suite. This talk will provide an overview of the system upgrade approach and the tools used to efficiently manage and test changes to both our data and software.

BACKGROUND INFORMATION

The purpose of the NIF is to continue research into nuclear fusion, specifically laser driven inertial confinement fusion (ICF). To accomplish this, NIF uses 192 lasers. When combined, the lasers can deliver up to 2.1 MJ of energy at 351 nm. For a sense of scale, the NIF building containing the lasers stands four stories tall and can fit three American football fields on the roof.

Responsibility for driving the NIF through its experiments lies with the Integrated Computer Control System (ICCS). At the lowest level, ICCS provides direct control of devices for manual operations during maintenance and troubleshooting tasks. On top of that, it provides multiple layers of automation which allow less than a dozen operators to configure and control more than 66,000 control points on over 2,300 processors and embedded controllers through the course of a shot cycle.

Behind the scenes ICCS has chosen a data driven architecture to keep things manageable. A predominantly Java code base of over 3 million lines of code provides the basic implementation of each control type, the hooks to communicate via Common Object Request Broker Architecture (CORBA) protocols, and the automation frameworks. Going along with this, an Oracle database stores all information needed to instantiate the control points, automation scripts, and the experiment configurations.

OTS LASER SYSTEM UPGRADE

As part of the NIF team's continued effort to improve our understanding of the plasma conditions, we recently deployed a diagnostic capable of measuring OTS from the imploding target. This method significantly improves the precision at which we can measure the plasma's temperature, density, and flow velocity. From this we'll gain a better understanding of how the laser interacts with the plasma, and how we can further reduce unwanted interactions. This in turn will lead to even better symmetry during the implosions and higher fusion yields. [1]

This upgrade consists of two large scale pieces: the OTS Laser (OTSL) and the OTS Diagnostic (OTSL-D). Unlike previous upgrades such as the Advanced Radiographic Capability (ARC), OTSL did not reuse any of the existing NIF beam path. Instead, we built a new room in NIF's switchyard 1 to house the front end and amplification components. The laser light follows a new beam path into the target area. Just outside the target chamber wall, we convert the laser light from the front end's 1053 nm to 211nm.

OTSL-D consists of two primary components. Frist you have the diagnostic package consisting of a spectrometer and alignment cameras. This package was specifically designed to work with either OTSL at 211 nm or a standard NIF 351 nm beamline as the probe laser. This allows us to install the diagnostic in multiple locations and inspect the target from multiple angles. In addition to the diagnostics, OTSL-D contains a separate set of alignment mirrors and cameras in a laser launch package. Due to the location where OTSL enters the target chamber, it cannot fire directly into the hohlraum's laser entrance holes. To get around this, we reflect OTSL off the laser launch mirrors and into the target.

OTS Software Upgrade Scope

To support the OTSL system, we needed to modify every layer of the ICCS system. Down at the Front-End Processor (FEP) layer, we made code changes to support multiple new device types, such as energy diagnostics,

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV021

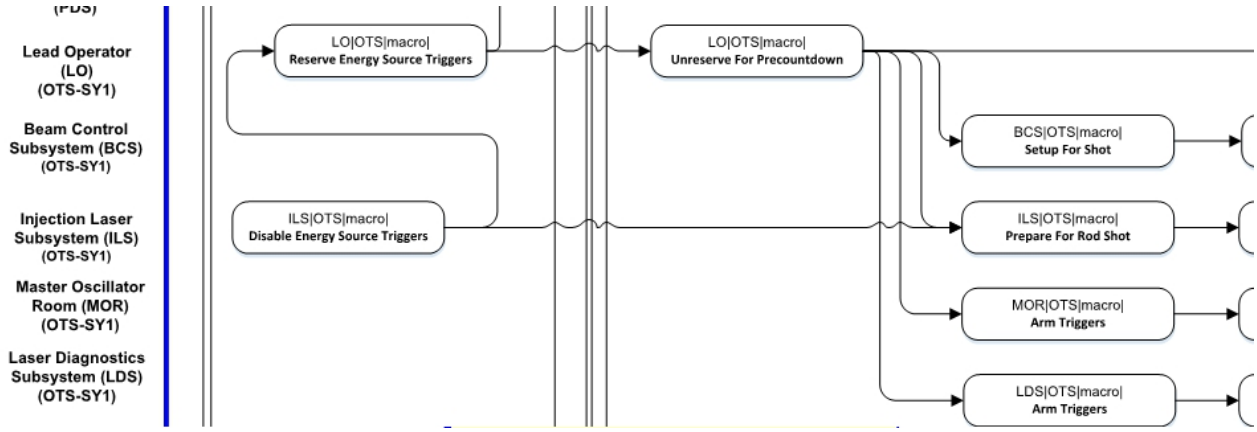

Figure 1: Snippet of OTSL shot automation swim lanes.

cameras, and power supplies. Along with the code changes, our database team needed to add new database schema and validations to support the new device types. Also in the FEP layer, we made large scale data changes to support additional instances of already supported devices.

In the next layer above, our automatic alignment team implemented a large number of new alignment loops. [2] While some of these loops were similar to existing NIF alignment loops, some operations (such as the tuning the 5ω converter crystals) required brand new approaches and image processing. Also in this layer, we needed to expand our pulse shaping system to support the new OTSL front end. Rounding this layer off, we needed to expand our status and propagation processes to handle all of the new control points.

The ICCS shot automation layer also saw significant changes. The ICCS shot team expanded their frameworks to handle this new class of diagnostic system. This resulted in 5 new automation swim lanes capable of setting up OTSL in parallel with the rest of NIF (Fig. 1). They also added multiple branching points to allow the possibility of OTSL-D being physically installed and ready for alignment operations at different points in the shot cycle. Finishing all of this off was another large data change specifying the exact details on when shot should position/verify/lock out every control point.

Last but definitely not least, multiple GUI changes were needed as part of this effort. Each of the new device types required a brand-new maintenance panel. In addition, we added new broadviews aligning with the OTSL control points diagram (Fig. 2). Several of our maintenance and commissioning tools were expanded to work with the new OTSL devices. Finally, we added tools to existing maintenance panels to assist with commissioning the system, such as real time image processing to our video displays.

MANAGING THE DATA CHANGES

Given the amount of data we deal with, the ICCS team has developed a number of tools to simplify manipulation of our databases. These tools range from very generic to highly specialized one time use tools. For the purposes of this paper, we'll be focusing on the generic tools that may provide inspiration to other teams.

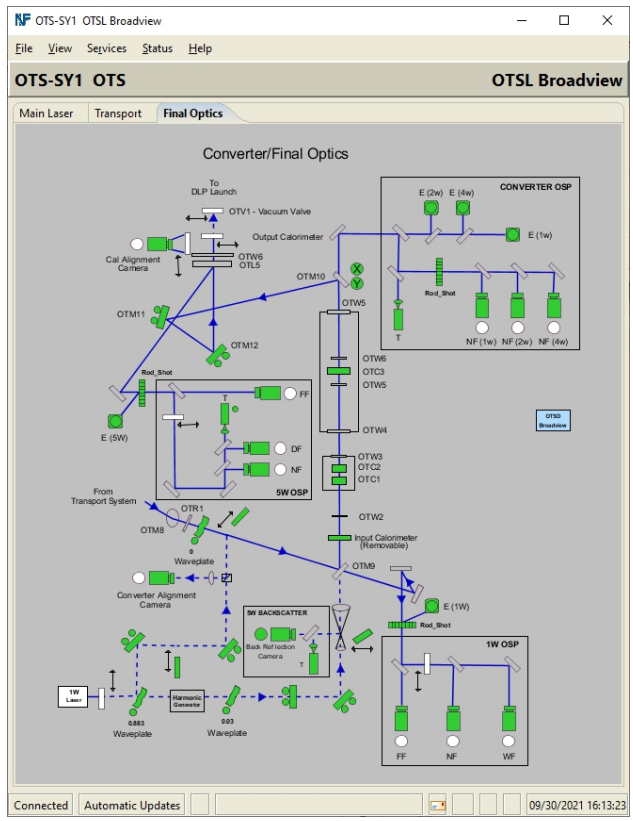

Figure 2: Broadview showing OTSL control points in the target area before entering the target chamber.

QuickMod

Originally introduced as a way for non-software developers to safely and efficiently make database changes, QuickMod rapidly became our standard method of making database changes. At the highest level, QuickMod translates Excel files into a SQL statements. Every sheet in the file can perform one of four basic operations (INSERT, UPDATE, DELETE, or CLONE)

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV021

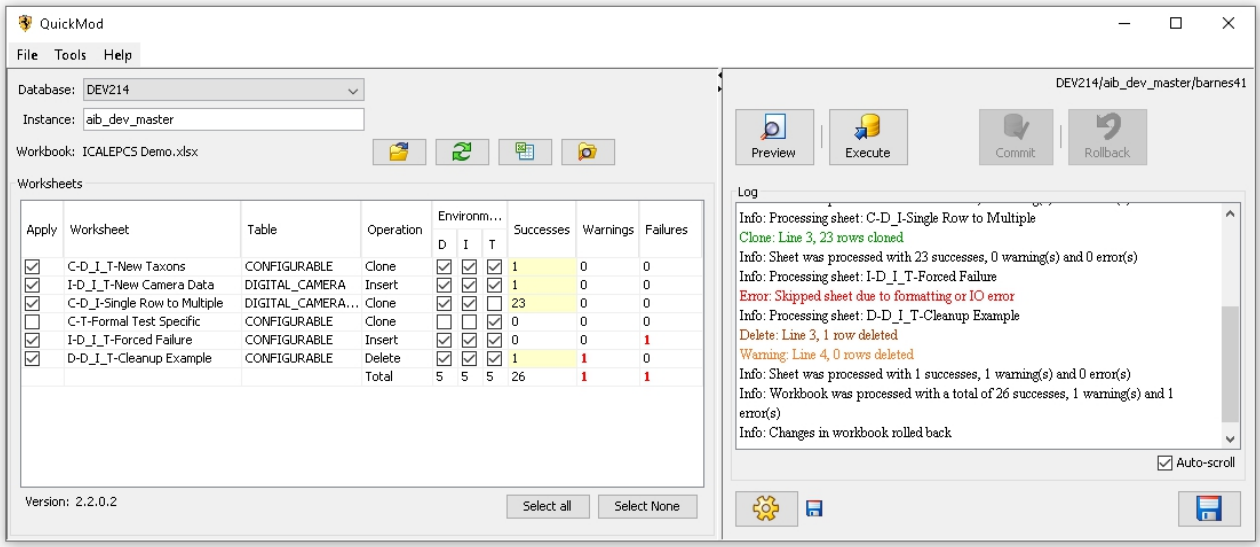

Figure 3. QuickMod GUI Interface.

on a single table. The first three operations align with the standard SQL definitions. Our clone operation combines multiple SQL statements to atomically query a table, modify the results, and insert the new rows back into the same table. QuickMod also evaluates Excel functions before applying each sheet. This allows for concise workbooks that can tailor themselves to each environment. For example, we commonly create workbooks that derive machine hostnames based upon the environment the data is being applied to (development, integration, formal test, or production). To round all this out, QuickMod provides a simple GUI interface (Fig. 3) for application of single files and a command line interface for release automation. The GUI interface provides users quick feedback on the success/failure of each operation and the ability to roll back changes before committing them to the database.

QuickMod has demonstrated several advantages. First, it provides an easy method for non-software developers to make database changes. We frequently have a software developer build an example Excel file for a subject matter expert (SME). The SME can then adjust values in the Excel file and have operations modify the database without further involvement of the software team. Second, the input files can easily be reviewed both before and after application to the database. Third, QuickMod provides a convenient place to implement validations and restrictions not easily performed by the database. For example, the ICCS architecture allows for multiple developers to work in their own personal instances while sharing the same database schema. This was done by adding an INSTANCE column to almost all database tables. Since developers rarely need to modify multiple instances at once, QuickMod restricts application of changes to a single instance at a time. Developers still have the option of falling back to SQL if they truly need to modify multiple instances simultaneously. Fourth, the granular feedback provided by the QuickMod GUI has reduced the time needed for developers to track down errors in their data changes. After executing, QuickMod provides the number of successful changes, warnings (e.g. an update that modified no rows), and errors on both a per sheet and per row basis. It also provides a log window with full details including the exact row/sheet that caused the error. Finally, by controlling the files in our configuration management system and deploying via command line automation, we ensure we deliver the entire set of intended changes to each environment.

While QuickMod is exceptionally useful tool, there are a few challenges that made it insufficient for a project the size of OTSL. The largest challenge comes from the domain knowledge needed to initially construct an input file. The ICCS configuration database contains over 1400 tables. Within each of those tables, we have a mix of columns that need to change with every entry and those that rarely change. Thus, developers frequently find themselves needing to consult domain experts both to simply identify the needed tables, as well as to identify what questions they need to ask of external teams. The next challenge comes from the quantity in data. This shows up in individual typos, copy and paste errors, as well as simply identifying when rows or sheets are missing.

Batch Cloning

To help deal with the challenges of using QuickMod alone, we created a batch cloning tool. For input, the tool takes in a list of existing control points, their new names, and new processor information. The tool first reads in all data related to the existing control points. It then adjusts the data based upon the new names and processors. Next, it removes any duplicate data. For example, when multiple digital input/outputs are defined on the same card/processor combination, it consolidates the data to a single definition of that card. As output, the tool generates an Excel file already formatted for use in QuickMod. In doing so, it highlights columns that developer likely needs to update after consulting external teams. Finally, the tool ensures that the output files will

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV021

create emulated devices in offline environments and real devices in production.

We ended up using the batch cloning tool for around ¾ of the OTSL data, and it quickly showed its benefits. With the first batch of OTSL control points, it cut the effort needed in half. And that time includes the effort to develop the tool. While it didn't completely eliminate the need to reach out to the domain expert for each device, it drastically reduced the amount of information needed. It also removed the possibility of missing data.

After using the tool, we also found that it introduced some new challenges of its own. Specifically, the tool did not make it easy to respond to requirements changes after the developer had started their work. The batch cloning tool excelled at getting the first 80% of the work done. After that, a developer needed to manually update the generated QuickMod file to include information gathered from other teams. Since all changes were consolidated into a single file, developers frequently would choose to manually respond to requirements changes instead of rerunning the tool and having to restart their work. The second challenge showed up when we started commissioning. Since the tool copied all data related to the source control points, it frequently copied too much data. Many of our control points have named configurations that we call setpoints. We found a number of instances where we deployed setpoints that either didn't make sense with the newly deployed control point or caused confusion since they were supposed to be created during commissioning.

Templates

Learning from our experiences with the batch cloning tool, as well as other data heavy project, the ICCS team choose to upgrade QuickMod to support templates. The templates are initially designed by a domain expert. They start with a definition sheet that contains both documentation on how to use the template and a section where each row equates to one new control point being generated when applied. After that, the Excel file contains all of the data necessary to make a single instance of a control point. Unlike a standard QuickMod input file, fields that we expect to change are replaced with template variables (such as <<name>>). When QuickMod applies the file, it will run all sheets after the definition sheet once for every new control point being added. And before each run, the template variables are replaced with the values defined in the definition sheet.

While this new system initially requires more time from a domain expert, it has shown several advantages. First, non-experts can use the templates with little to no input from the domain expert. Frequently the documentation provides enough guidance on what questions to ask other teams, and which columns the developers need to worry about changing. Second, a developer can easily respond to a change in requirements. For example, changing a control point's type frequently boils down to removing a row from one definition sheet and adding a new row in different template. Third, since the templates are essentially the data for a single instance, they become easy to generate and review. In ICCS, most of our control types have data in less than ten tables, and frequently only have one to four rows per table.

TESTING THE CHANGES

For testing changes of this magnitude, ICCS both reused existing testing frameworks (both automated and manual) and adapted some of our ancillary tools to assist in testing. This allowed us to deploy the software with high confidence that minimal adjustments would be needed in production. It also allowed us to rapidly make significant code changes when the new energy diagnostics were significantly upgraded after our initial software deployment.

Testing Device Control Changes

Our front-end processor (FEP) layer is responsible for making any piece of hardware available via a CORBA interface. We have two automated test frameworks responsible for unit and integration testing this layer. Both frameworks are built on top of JUnit and tied into our nightly build system. We designed the unit test framework to minimize the effort needed add new tests. Developers simply need to add a single line annotation to either the class or to individual test functions in order to designate a test suite. When the test framework runs, it scans the compiled code base for these annotations and dynamically builds up the test suites to run. We currently have two test suites fast and slow which respectively take approximately 2 minutes and 10 minutes to run. These tests predominantly focus on validating individual Java class.

The second framework performs black box testing via the CORBA communications layer. We introduced this framework when we ported the FEP layer to Java [3] and continue to expand it with every new control type. Given that a single CORBA interface frequently supports multiple different models of hardware (e.g. different cameras), we needed the ability to rerun the same test classes on different control points in our system. To support this, we choose to define the test suites in our configuration database. We then created a custom JUnit test runner that ties into a running instance of our system and queries the database to determine the tests that need to run. The runner also gives us the capability to run multiple test suites in parallel under different Java virtual machines. In addition to the test runner, we created a few JUnit rules to assist with common low-level operations (such as locking out control points). The framework also provides utility/base classes for testing areas where the functionality is relatively similar across control types (e.g. our interfaces for setpoints/named configurations). For a sense of scale, the test suites in this framework take between 5 minutes and 6 hours to run.

Testing Shot Automation Changes

In our offline integration environment, we predominantly rely on nightly runs of our automated shot

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV021

tester (AST) [3]. We currently have AST run five shots in each of our three releases under active development. To maximize our testing coverage, the experiments are specifically designed to have a mix of capabilities (such as OTSL). With each of these shots, AST will do its best to drive each shot to completion and provide a report on any failures 15 minutes before by our morning meeting. AST also provides us the capability to run shots on demand throughout the day.

In addition to AST testing, we perform a large variety of manual tests on shot changes. During integration testing of new functionality, developers will manually check all shot operations (e.g. positioning control points or verifying positions). In formal test, our QA will run multiple manual shots to confirm expected operations. They will explicitly test any verifications responsible for machine safety on a new or modified component (a different system is responsible for personnel safety checks). After deployment to production, we perform a final round of testing during the initial commissioning test shots. This testing includes manual verification that the automation layer correctly positioned all new control points, and a final verification of all machine safety checks.

COMMISSIONING THE SYSTEM

Due to the amount of new hardware with OTSL, we needed to come with a way up with a commission and tune the system without interfering with NIF's ability to continue experiments. We initially started by providing the OTSL team with an interface into the control system that could only access the OTSL control points. The team was then required to manually setup and fire OTSL. After attempting a few shots in this manner, we quickly realized the need to partially automate the process. To do this, we made use of our Target Alignment Assistant Tool (TAAT) [4]. TAAT was originally designed to allow rapid automation of alignment procedures for unique targets. As such, it reads in specially designed Excel files that define the scripted actions to take, where to ask the user for input, and where to branch if necessary. It then provides a basic user interface while it sends the corresponding CORBA commands into the system. With this, we were able to reduce the time to manually arm and archive OTSL devices during each iteration of a tuning shot from 1 hour to 5 minutes.

COMMISSIONING THROUGH COVID

With the ICCS team working almost exclusively from home, the pandemic pushed us to rethink and develop new strategies for supporting remote troubleshooting and commissioning operations. One of the strategies has worked out so well that we plan to continue using it even after the team returns on site. When a team working on NIF needs assistance, they initially setup a video teleconference and reach out to the domain experts needed. As part of the teleconference, the operations team will share the desktop of the console where assistance is needed. This allows the experts to simultaneously see what's going on in the control room, have a meeting with everyone involved, and still access tools that they have on their development machines. Building on top of this, the ICCS team maintains a Microsoft Teams channel providing summaries and status updates for any unplanned assistance requests.

The remote desktop capabilities have proven exceptionally useful in many ways. The most significant improvement is that we can have experts see the operators' screens and no longer need to rely on their descriptions of what they're seeing. This both reduces confusion and allows experts to notice small items that the operators may have overlooked or assumed were normal. This has also proven extremely useful for the alignment team, as they can now watch the alignment cameras in real time. Additionally, this functionality allows developers to remotely troubleshoot problems in labs that previously would have required them to don protective equipment and operate under escort due to safety hazards.

The strategy has also made the team more effective both when commissioning and troubleshooting operations outside of normal hours. The easy availability of teleconferencing has significantly reduced instances of multiple groups working on the same problem in parallel in their own communications silos. The Teams channel has provided management an easy way to stay appraised of problems in production. It has also improved the details available for deep dive troubleshooting the next day. Finally, the ability to work effectively from home has been a boon for SMEs that need to support shot operations well outside of their normal hours, or for extended duration shots.

CONCLUSION

The ICCS team successfully leveraged custom data manipulation tools, automated testing, and a remote work strategy to efficiently deliver controls for the OTSL system. For other teams considering similar tools and strategies, we have a few recommendations. First, focus on keeping each iteration of a tool/process change small and focused. This will lead to tools being completed faster and easier. It also keeps you from sinking significant effort on a tool that might not fully meet your needs. Second, always consider making tools more generic to allow for reuse in unforeseen circumstances. For example, when we initially wrote TAAT, we had no inkling that we would use it to automate non-alignment activities. Third, always keep an eye out for automation opportunities, especially testing.

ACKNOWLEDGEMENT

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

ISBN: 978-3-95450-221-9 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2021-MOPV021

REFERENCES

[1] P. S. Datte *et al.*, "The design of optical Thomson scattering diagnostic for the National Ignition Facility," *Rev. Sci. Instrum.*, vol. 87, p. 11E549, Nov. 2016. doi:10.1063/1.4962043

[2] A. Awwal *et al.*, "Image Processing Alignment Algorithms for the Optical Thomson Scattering Laser at the National Ignition Facility," presented at ICALEPCS'21, Shanghai, China, paper WEAL01, this conference.

[3] B. T. Fishler *et al.*, "Sustaining the National Ignition Facility (NIF) Integrated Computer Control System (ICCS) over its Thirty Year Lifespan," *in Proc. ICALEPCS'17*, Barcelona, Spain, Oct. 2017, pp. 1201-1205. doi:10.18429/JACoW-ICALEPCS2017-THCPL06

[4] M. Fedorov *et al.,* "New Visual Alignment Sequencer Tool Improves Efficiency of Shot Operations at the National Ignition Facility (NIF)," *in Proc. ICALEPCS'17,* Barcelona, Spain, Oct 2017, pp. 328-333. doi:10.18429/JACoW-ICALEPCS2017-TUMPA01