# CODAC CORE SYSTEM, THE ITER SOFTWARE DISTRIBUTION FOR I&C

Franck Di Maio, Lana Abadie, Changseung Kim, Kirti Mahajan, Denis Stepanov, Nadine Utzel
ITER Organization, Route de Vinon sur Verdon, 13115 St Paul Lez Durance, France

*Abstract*

In order to support the adoption of the ITER standards for the Instrumentation & Control (I&C) and to prepare for the integration of the plant systems I&C developed by many distributed suppliers, the ITER Organization is providing the I&C developers with a software distribution named CODAC Core System.

This software has been released as incremental versions since 2010, starting from preliminary releases and with stable versions since 2012. It includes the operating system, the EPICS control framework and the tools required to develop and test the software for the controllers, central servers and operator terminals.

Some components have been adopted from the EPICS community and adapted to the ITER needs, in collaboration with the other users. This is the case for the CODAC services for operation, such as operator HMI, alarms or archives.

Other components have been developed specifically for the ITER project. This applies to the Self-Description Data configuration tools.

This software has also been used for the production of the first I&C applications in Cadarache for the monitoring of power stations.

This paper describes the current version of the software as released in 2013 with details on the components and on the process for its development, distribution and support.

## INTRODUCTION

The ITER control system is composed of central control services that will be installed at the ITER site in Cadarache and of the plant systems controllers that will be, for most of them supplied with the plant systems [1] [2].

For enhancing homogeneity and preparing for the integration, the ITER Organization is developing, distributing and supporting a software framework based on EPICS to be used for the development and tests of the control software of the plant systems. The same framework is used for configuring the on-site systems (servers, operator stations). It is supporting all the standards adopted for the ITER controls and aim at facilitating their adoption by the partners. The framework and the associated tools are packaged into the ITER CODAC Core System distribution.

For the developments and tests at production sites, the central servers are replaced by a local computer, named Mini-CODAC. The CODAC Core system includes a reduced version of the central services to be installed on the Mini-CODAC systems.

The distribution is continuously updated for adding new features and for fixing issues detected by users, user support and testers. These changes are packaged into regular releases issued twice a year since 2010. The operating system, Red-Hat Enterprise Linux (RHEL), is part of the distribution and can be updated once a year.

## COMPONENTS

### *Configuration Toolkit (SDD)*

A dedicated suite of tools, named the SDD toolkit [3], has been developed for the project. It allows developing the detailed design of the local controls using editors and generating the configuration files required for building the control software and for configuring the CODAC services.

All the configuration data are stored into a local database. Synchronization tools allow exchanging data between local and central databases

The SDD toolkit is mandatory for any development that will be delivered to ITER. Verifications are made during the local design against ITER conventions and shared data (ex: plant breakdown structure).

The toolkit allows creating and configuring all EPICS variables, the interface variables on PLCs and the data exchanged for plasma control. Alarms configuration and archiving parameters are also part of the configuration.

The configuration data includes each controller and the I/O modules the software is controlling. The signals are also defined with their links with the variables and, when required by the software, with the connection to the I/O modules.

The user can edit files for updating list of variables, list of signals or EPICS variables declaration but the format is imposed by the toolkit and the files shall be parsed and the content imported back into the database.

The latest extensions are for configuring the interface of real-time programs with the plasma control network, as described below.

Following the progress of the plant systems design, verifications of centrally available data against local design will be enhanced. As an example, the list of variables that are centrally defined to constitute a system interface should be checked against the variable defined locally.

### *Build Tools*

Using SDD configuration tools, the users develop projects that implement the control software whose scope can be a complete plant system or parts of one or many systems. Such a project is usually a deliverable for a

ISBN 978-3-95450-139-7

Procurement Arrangement with one ITER Domestic Agency or for a contract with a supplier of the ITER Organization. The configuration data of such projects should be submitted to a central repository as part of a delivery.

The project's configuration data is used by the developers for building and testing the software. Additional files are produced by the developers as required. These can be C/C++ programs, SNL programs (the EPICS State Notation Language), operator interfaces and plot files for which tools are included in the Core System distribution.

Each project is associated with a software unit in the ITER central source repository in which the source files shall be delivered. The software packages to be deployed on the operational systems are built from the files in the source repository by automatic processes.

A unique build process, implemented with the Apache Maven framework, is used on local development systems and on central build servers.

The main steps in the process are the compilation that produces the executable files as well as other run-time files and the packaging that produces the RPM (Red-Hat Package Manager) files for installation. The execution of the unit tests is an intermediate step before packaging. After compilation, the software can be started on test systems. On operational systems, the software can only be activated by having it installed.

In the latest versions of the CODAC Core System, a dedicated editor implements all build commands and these commands haven been also integrated to the SDD tools. As a result, the developer can design, build, test and package the software from the same tool, as illustrated in Figure 1 below. Command line interface, as present in previous versions is also available.

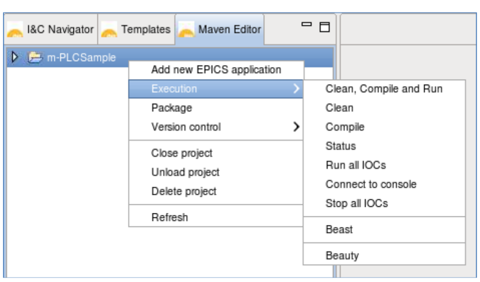


Figure 1: Build commands embedded in SDD editor.

### *Integration of PLC*

The integration of PLCs (Siemens S7) is a basic feature implemented from the earliest versions. The interface between the PLC and the central controls is implemented by variables that are part of the I&C design and that are generated both for the PLC and on the Linux system interfacing it as EPICS variables [4].

The PLC control software must import the generated variables and use these as the interface with the control system. The communication functions in PLC and in the EPICS Linux process are included in the generated files.

The implementation of this integration has been rather complete and stable since 2012. The extensions made by ITER are for handling communication errors, detecting inconsistent configurations and producing health information.

The latest releases of the software distribution have been extended with the support for redundant configurations.

### *Integration of I/O Devices*

Currently, the only recommended hardware platform for ITER controllers is rack-mounted PC driving PXIe crates and a list of PXI/PXIe modules, from National Instruments, has been selected for building local controls.

A small set of PXI I/O modules is supported by the tools distributed in the CODAC Core System: a general purpose board, PXI-6259 (AO, AI, DIO, counters), a digital I/O board with signals isolation and two versions (old and new) of a timing board, PXI-6682 and PXI-6683H.

For these boards, the configuration tools encapsulate the configuration of the associated EPICS variables to allow the user to configure the software be defining links between variable and board channels or by defining board specific configuration parameters used in templates.

High performances boards (PXIe-6368 and FlexRIO) can also be used with ITER supplied software (Linux drivers) but are not fully supported (i.e. not handled by the user support).

### *Development of Real-time Programs*

From the latest versions, the user can add programs (C++ language) in its project and have these handled by the build process.

For plasma control, the participating nodes that are acquiring data, executing computation or executing control are exchanging data over a dedicated network named Synchronous Databus Network (SDN). This network is based on Ethernet using multicast UDP.

The software components for SDN communications have been added in the 2013 releases. The data packets, named SDN topics, are designed with the configuration tools (SDD) and the configuration of a real-time program shall include the SDN topics the program produces or consumes. The SDD toolkit also configures the program files for using publish/subscribe functions of the declared SDN topics.

Fast controllers shall all be connected on the Timing Communication Network (TCN), which is an IEEE-1588 enabled Ethernet network that distributes the ITER time. The PXI timing modules are connected to this network and it is also foreseen that computers will be connected using built-in PTP compliant network interfaces once this becomes supported in the standard RHEL distributions. In either case, a daemon process is updating the system time from the received ITER time.

One more service with its dedicated network need to be supported in future CODAC Core System versions for the archiving of scientific data or other large data.



ISBN 978-3-95450-139-7

### Control System Studio

The standard common services for operators such as operator graphical interfaces, alarms and data arching are implemented by tools developed within the EPICS community using the Control System Studio framework.

These tools (BOY, BEAST, BEAUTY…) are shared with the other users, ITER contributions being incorporated with the other ones into a unique repository. They are continuously enhanced by the development teams. As a result, on every CODAC Core System releases, many features are added [5].

In the latest releases, ITER has extended the alarms handling with notification modules for executing actions such as mail or SMS emission upon alarm occurrence. In parallel, web interfaces has been completed so that OPI, alarm views and data plots can all be used from web browsers.

The operator tools will now be extended for the preparation and execution of the acceptance tests of the plant systems,

## MANAGEMENT OF THE DISTRIBUTION

### Releases

The CODAC Core System distribution has been developed and released at regular milestones every year since 2010. A major release has been issued in February and a minor one in June or July. Versions are identified with three identifiers: major_id.minor_id.maintenance_id. The more recent version at the time of publication is 4.1.0, released in July 2013.

The version number is reflecting the level of changes. Changes in the infrastructure, such as new operating system version, changes in the EPICS base or changes in the structure of the configuration database require a new major version while the release of a new maintenance version is restricted to packaging a set of bug fixes without new features.

A patch production mechanism has also been added for providing a user, or a controlled set of users, with a change of limited scope that resolve a recorded issue.

It is planned to continue releasing new versions at fixed dates. However, it is foreseen to release more minor versions than major ones for facilitating application migration and Core System upgrades.

### Milestones

As mentioned above, two releases are planned every year with fixed target dates. The list of milestones is derived from these dates incorporating the production of preliminary versions for tests. The tasks before the release include verification and approval of the code, the tests and the documentation. The tasks after a release include updating the training and the public distribution.

The candidate features for the next year releases are defined before end of August. From September to December one beta version is produced every month from a code freeze on the last day of the month. Tests are executed on each beta version, according to the changes included. From January, beta releases become release candidates and tests shall be stable. New versions can be produced up to once a week until the quality is good enough for a release. Quality evaluation is made from the test reports and from the severity of unresolved issue. For reaching the target date, new features or new components can be de-scoped from the release.

For the mid-year release (June/July), the process is reduced with only two monthly beta versions instead of four. It is also expected that the scope of the changes is reduced and mainly includes extensions and fixes that could not fit in the main release or that have been postponed.

### Issue Tracking and Change Management

Changes are managed by means of the Bugzilla issue tracking system with a dedicated process to record decision on changes and to follow their execution.

New requests for bug fixes or extension are entered by the developers, testers and support staff. Each entry is recorded as a "bug" associated with a software component and is either defined as an extension or qualified with a severity.

A coordination body is reviewing new entries as well as those requiring action (schedule issue or argued resolution). Each software component is managed by its owner, responsible for the quality of any bug associated with this component. The coordination body endorses assignments of new bugs for resolution with a designated assignee. Alternatively the quality responsible of a bug can endorse its assignment.

When assigning a bug, a target version of the CODAC Core System is also set. This defines the target date for resolution as well at the SVN branch the assignee can and shall use. The recommended method is to generate comments in the bug automatically when committing a change in SVN. Assignee is in charge of indicating the resolution of the bug when done. The quality responsible should afterwards confirm the resolution by indicating that the bug resolution has been verified. The recommended verification for a significant change is to have it covered by the tests associated with the software component and defined in its test plan.

The bug data base is a reference for the user support but is not advertised to the end users.

The list of changes described in Bugzilla is rather exhaustive. The numbers of issues qualified as "fixed" in the last two years are indicated in Table 1 below.

Table 1 – Fixed Issues per Release

| Release | Fixed issues |
|---|---|
| 3.0.0 (2012-02-15) | 468 |
| 3.1.0 (2012-06-22) | 305 |
| 4.0.0 (2013-02-15) | 428 |
| 4.1.0 (2013-06-22) | 411 |



ISBN 978-3-95450-139-7

### *Tests*

The development and execution of tests became a big task from 2011. The work of many teams working in parallel needs to be verified during a short period and undetected errors in a released version had very bad consequences on the support and on the development tasks for providing fixes.

For each software component, a test plan defines all the tests, gives the test coverage against requirements and bugs and shall provide non-expert testers with the instructions for the test execution. The test plan is part of the delivery with the code and the documentation.

Before releasing a new version, all test reports should have been approved. These reports are input for quality evaluation abut are also providing the support team with a reference for known issues and performance values of an official version.

Integrated tests are developed and executed by the support team. It is foreseen to extend these with dedicated test beds and to have more tests made at the ITER site.

Regression detection is quite good now while tests are usually updated from the recorded major bugs, so verification can be made on all versions after the fix.

In theory, backward incompatibilities can be detected by executing the final tests on the previous version of the software but this has not been executed in a systematic manner up to now.

## USER SUPPORT

Organizations that have a link with the project are registered in a distribution server to allow an administrator within the organization to configure local systems from the ITER distributions. Users belonging to this organization can also request assistance from the CODAC support. In September 2013, 57 organizations were registered.

A dedicated team is in charge of addressing all requests made to the support from registered users. The Bugzilla issue tracking system is also used for the follow-up of the user requests and for involving experts when required. Many requests are for assistance in the installation or upgrade of the CODAC Core System distribution but the requests from developers are now increasing.

The earlier version of the CODAC Core system issued in 2010 (v1.x) and in 2011 (v2.x) are not anymore within scope of the support, except for assistance to migration. Reducing the number of old releases that have limitations and bugs is a worry because contracts may have been concluded with the specification of an old version. Many efforts have been made in migration support with tools and documentation in the last two years but dedicated assistance is often required for helping users to migrate their application to a new version of the framework. From the version 3.0 (issued in February 2011) in which a new major release of RHEL was introduced (6.1), a complete re-installation is not anymore required for upgrading the CODAC Core System to a new version. Still, automated upgrade is quite difficult to achieve and is not completely fail safe.

Since 2011, hands-on workshops are organized at ITER premises after each release with presentation and exercises. While controllers are used for such exercises, assistance is limited. Around 100 people have attended such workshops already.

The number of on-site training sessions (about five a year) will not increase but these will be completed with on-line self training.

## CONCLUSIONS

The development, distribution and support of a common framework and a common set of tools for the development of the plant system controls remain big tasks at ITER. The configuration tools are very exhaustive now and they allow a large part of the design to be made with graphical tools for the development of the software at the production sites. And the detailed design of the plant systems software, saved in relational databases, can be transmitted to ITER before integration.

Build tools are very complete now with the same workflow from source files and configuration data at the developer sites and on integration servers at the ITER site.

Current developments are now mainly focused on the support for real time control to comply with the project schedule.

During the last period, efforts were made to generalize test procedures and test execution for improving quality and reducing the cost of the support.

The views and opinions expressed herein do not necessarily reflect those of the ITER Organization

## REFERENCES

[1] A. Wallander et al. “Baseline Architecture of ITER control system”, IEEE TNS vol. 58 1433-1438, Aug 2011.

[2] A. Wallander et al., “Approaching Final Design of ITER Control System”, TUCOAAB03, ICALEPCS, 2013.

[3] L. Abadie et al., “SDD toolkit: ITER CODAC platform for configuration and development”, TUPPC003, ICALEPCS 2013.

[4] S. Pande et al., “CODAC Standardisation of PLC Communication”, THPPC004, ICALEPCS 2013.

[5] N. Utzel et al., “ITER Contribution to Control System Studio (CSS) Development Effort”, TUMIB08, ICALEPCS 2013.



ISBN 978-3-95450-139-7