Fusion Engineering and Design 87 (2012) 2213–2217

Contents lists available at SciVerse ScienceDirect

Fusion Engineering and Design

journal homepage: www.elsevier.com/locate/fusengdes

The self-description data configuration model

Lana Abadie*, Franck Di Maio, Wolf-Dieter Klotz, Kirti Mahajan, Denis Stepanov, Nadine Utzel, Anders Wallander

ITER Organization, Route de vinon sur Verdon, 13115 St Paul Lez Durance, France

HIGHLIGHTS

- We use the relational model to represent the configuration data for ITER.
- We explain the different modeled views namely physical, functional and control.
- We explain how this information is used to generate configuration files.
- We explain that this information is validated.

ARTICLE INFO

Article history:
Available online 2 November 2012

Keywords:
Databases
System configuration
Spring, Hibernate

ABSTRACT

ITER will consist of roughly 160 plant systems I&C delivered in kind which need to be integrated into the ITER control infrastructure. To make the integration of all these plant systems I&C, a smooth operation, the CODAC (Controls, Data Access & Communications) group release every year the core software environment which consists of many applications. In this paper we would like to describe what configuration data and how it is modeled in the version 2. The model is based on three views, the physical one which lists the components with their signals, the functional view which describes the control functions and variables required to implement them and the control view which links the two previous views. We use Hibernate as an ORM (Object Relational Mapping) framework with a PostgreSQL database and Spring as a framework to handle transactions.

1. Introduction

CODAC will integrate and control around 160 plant systems I&C. To allow a smooth integration, each year, we release a new version of the core software environment which follows the PCDH (Plant Control Design Handbook) [1]. The main purpose of the PCDH is to provide guidelines and standards to plant system I&Cs designers.

1.1. EPICS

CODAC selected EPICS [2] as a conventional control framework. EPICS uses PVs (Process Variables) to control a system; it is not device-oriented. PVs are visible and can be accessible between systems (including CODAC). A PV can be viewed as a C-structure. There is no hierarchy between PVs, but PVs can be linked with each other. This PV linkage is a key feature of EPICS. It can be considered as a graph where nodes are PVs and edges are links.

The PV name has to be unique in the ITER control system. A PV has a record type like bi (binary input), bo (binary output), ai (analog input), etc. Another EPICS concept is the IOC (Input/Output Controller). An IOC can be viewed as a server. It initializes the drivers to interact with I/O boards. It loads in memory its PVs (similar to a server). A PV has to be loaded by one and only IOC, otherwise collisions with PV names will occur (Fig. 1).

1.2. Core software environment

The core software environment is based on Red Hat 64-bit. It includes several RPM (Package Manager for Linux systems) packages:

- EPICS device support and Linux drivers for boards listed in PCDH.
- S7 PLC support.
- Plant system I&Cs configuration handled by SDD (self-description data).
- HMI (Human Machine Interface) builder.
- Alarm and archive services.
- Utilities to build and manage EPICS projects based on Maven [3].

* Corresponding author.
E-mail address: lana.abadie@iter.org (L. Abadie).

0920-3796/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.fusengdes.2012.09.018

```
record (bi,"UTIL-EB20-S15:CR1-YT1")
{
        field(DESC, "clean contact/normal open")
        field(PINI, "YES")
        field(DTYP, "S7plc")
        field(INP, "@null_cfg/58 T=UINT8 B=0")
        field(SCAN, "I/O Intr")
        field(ONAM, "ON")
        field(OSV, "NO_ALARM")
        field(ZNAM, "OFF")
        field(ZSV, "NO ALARM")

}
```

Fig. 1. An example of PV. Its record type is bi (binary input can be also called digital input) and has several fields. DESC stands for description. PINI indicates if the PV has to be processed at IOC start up. Its two-states are described via ONAM and ZNAM fields. So this PV is ON state when it gets value 1 and is in OFF state when its value is 0. OSV and ZSV specify the alarm severity in case the PV goes in one state. In this example, whether the state is ON or OFF, no alarm is generated. The DTYP field indicates the type of the driver. Here S7plc means that this PV reads data from a S7plc. The INP field is mapped to a PLC datablock (the one being read). The SCAN field indicates how the data has to be read. Here the data will be read each time that an I/O interrupt occurs.

A major release is made on a yearly basis. In this paper, we will focus on the configuration of plant system I&Cs.

1.3. Static configuration

The self-description describes the static configuration for the plant systems I&C. It covers many topics going from documentation to list of signals. The self-description data can change only during maintenance phase. It does not cover dynamic information such as exclusion/inclusion of a plant system I&C, initial values before the start of a pulse, etc. In core software, the main objective of the self-description is to offer to the plant system designer/developer a toolkit to model its system in terms of PVs.

The toolkit has also to be able to automatically generate configuration files.

1.4. SDD Workflow

The SDD workflow is described in Fig. 2. It is split into two parts, central (at IO) and local (outside IO, all institutes/companies which have to develop I&Cs for DAs in the frame of procurement arrangements). There are data which shall come from the central database and replicated locally, namely the list of components and signals as they are defined during design engineering phase (link 1). Then, at local place the plant system I&C designer uses the SDD toolkit to configure the plant system I&C he/she is responsible of (link 2). Then once the configuration done, configuration files are generated (link 3). Then the plant system I&C programmer uses these files as a starting point and add the plant system logic into the different control components (links 4 and 5). The link 6 shows the type of data exchanged between the plant system I&C and the database (signals and I&C variables mainly). The plant system (PS) operator can act on the HMIs and reports problem to the PS I&C designer in case of bad settings (link 7). Then the final configuration gets delivered at ITER site (link 8) and gets verified and updated at ITER site for integration purposes (link 9).

2. SDD relation model

The configuration data has been modeled using the relational model. In the next sections, we are describing the different parts of the SDD model.

2.1. Objectives of the model

The main objective of the SDD toolkit is to generate automatically the configuration files for CODAC core services. In other words,

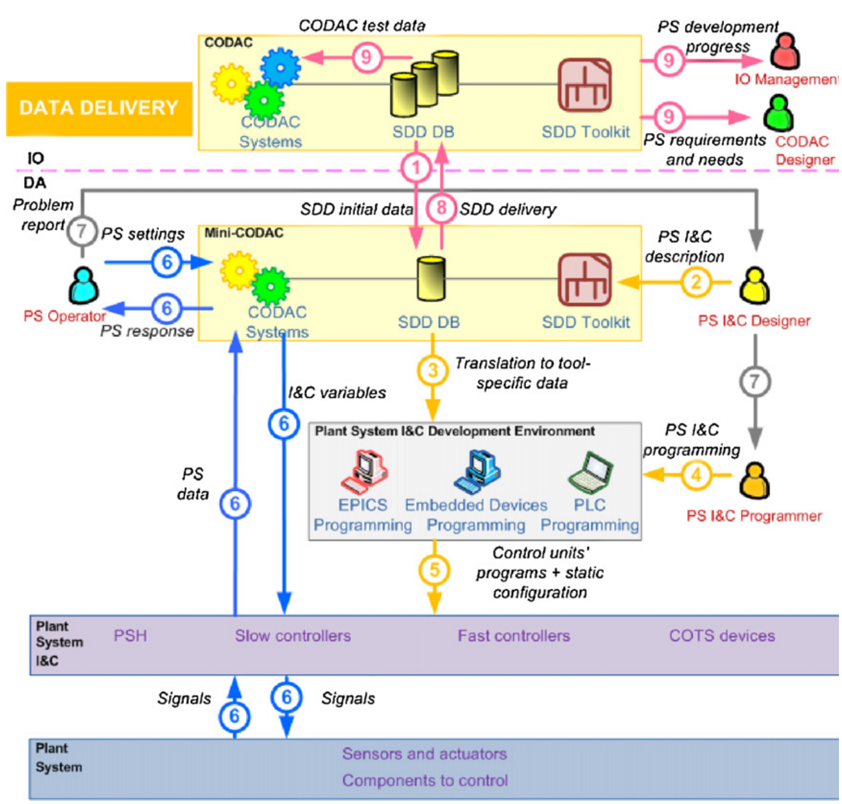

Fig. 2. SDD workflow.