G Model
FUSION-8712; No. of Pages 4
ARTICLE IN PRESS
Fusion Engineering and Design xxx (2016) xxx–xxx
Contents lists available at ScienceDirect
Fusion Engineering and Design
journal homepage: www.elsevier.com/locate/fusengdes

Fast development of real-time applications using MDSplus and MARTe frameworks

G. Manduchi[a,*], T.W. Fredian[b], J.A. Stillerman[b], A. Neto[c], F. Sartori[c]

[a] *Consorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), Padova, Italy*
[b] *Massachusetts Institute of Technology, 175 Albany Street, Cambridge, MA 02139, USA*
[c] *Fusion for Energy, Barcelona, Spain*

HIGHLIGHTS

- The paper describes the integration of two different frameworks for control and data acquisition.
- It describes the way the two frameworks have been integrated.
- It describes the advantages of this combined approach.
- It presents a case study of the utilization of the two integrated frameworks.

ARTICLE INFO

Article history:
Received 20 May 2015
Received in revised form 14 January 2016
Accepted 8 April 2016
Available online xxx

Keywords:
Real-time control
Data acquisition
Software frameworks
Software integration

ABSTRACT

The recent long lasting fusion experiments introduced a change in paradigm for control and data acquisition. While formerly implemented by different systems, using different software and hardware solutions, new requirements, such as the need of handling a sustained data stream, the availability of powerful general-purpose computers and the evolution of Linux towards real-time responsiveness make an integrated solution nowadays feasible. In the fusion community several frameworks have been developed for control and data acquisition and some of them are shared among different experiments. In particular, MDSplus represents the most used framework for data acquisition and management and MARTe is a framework for real-time applications originally developed at JET, but then adopted in several other experiments. Neither system can be used alone to provide integrated real-time control and data acquisition but, since their functionality complements, they can be used in conjunction. To achieve this, an additional layer has been developed so that data handled in real-time by MARTe components can be redirected to pulse file for storage. At the same time, configuration data, typically defined in the MDSplus experiment model, can be seamlessly transferred to MARTe GAMs during system configuration.

1. Introduction

In former experiments, the short duration of the plasma discharge permitted the use of transient recorders to take signal snapshots during the discharge. The data acquisition system was not actively involved during the discharge itself, and only in a subsequent phase provided the readout of transient recorders memory and the storage of acquired signals in a database. On the other side, the management of data streams was required for real-time control, but the limitations in computer systems performance restricted the applicability of general solutions, leading to specialized solutions for real-time control, different both in hardware and in software from the data acquisition system.

Modern experiments are characterized by an extensive use of real-time control components. In addition, larger devices with long lasting plasma discharge require the management of streaming data acquisition so that information is available during the discharge itself. Even in this case, however, different solutions are required to manage data acquisition and real-time control. Considering Analog to Digital (ADC) devices, larger data buffers are required to improve data throughput in data acquisition, but smaller data buffers are required when ADC devices acquire sensor signals to be used in real-time control in order to reduce latency. Small data buffers are required in data acquisition for real-time control to account for occasional missed deadlines in the cycle time of the control system. The same dichotomy (large buffers, large

* Corresponding author.
E-mail address: gabriele.manduchi@igi.cnr.it (G. Manduchi).

http://dx.doi.org/10.1016/j.fusengdes.2016.04.014
0920-3796/© 2016 Published by Elsevier B.V.

throughput vs. smaller buffers, smaller latency) holds also considering the hardware and software components of the computer systems used for data acquisition and real-time control, respectively.

While the technologies differ, the user's perspective of data in modern experiments is towards unification between data acquisition and control since the latter is becoming more and more an integrated component of the fusion device. Therefore data should be presented in the same way regardless of their origin and they may represent physical acquired signals as well as values computed by real-time control components.

The co-existence of different solutions sharing the same computer system hardware is made possible nowadays thanks to the rapid evolution of multicore computers that allow partitioning the computer into independent components sharing data in memory. The availability of multicore support with real-time responsiveness in Linux allows reserving a set of available cores for real-time operations without interfering with the rest of the system handling data acquisition and supervision [1].

Different frameworks have been developed in the fusion community to address streamed data acquisition and real-time control, using different techniques to increase data throughput in the former and to reduce latency in the latter and a fully integrated system could be built by integrating two frameworks addressing the two different functions. If we compare it to a new implementation from scratch, this approach has the clear advantage of retaining the experience and solutions gained in years of development and operation and ensures therefore both better availability and reduced time to target. In the presented solution, MDSplus [2,3] and MARTe [4,5], two frameworks used in fusion community for data acquisition and real-time control, respectively, have been integrated to provide a flexible and powerful framework which can be adapted to laboratory experiments running on a single machine as well as to large and distributed systems. The integration of MARTe and MDSplus has been first introduced in [6]. In that paper the advantages of a combined approach have been presented. This paper provides more insight into the way the two systems interact, and presents, as a proof of concept, the first application integrating such frameworks.

2. MARTe and MDSplus

The MARTe framework for real-time applications has been developed at JET and is currently in use also in other machines (RFX, FTU, Compass). MARTe is not the only available framework for real-time control and other solutions are adopted elsewhere, such as DCS in use at ASDEX [7] and foreseen in WEST, and PCS in use at DIID, EAST and KSTAR [8]. The main architectural design choices of MARTe are:

– The abstraction of the underlying platform, which makes the system portable to other non-Linux systems.
– The component abstraction which clearly separates the management of data flow from control computation.

MARTe is implemented in C++ and its modular organization is reflected in the underlying C++ class hierarchy. The integration of new components is carried out by implementing classes that extend a set of given interfaces (abstract classes). These interfaces represent the only required interaction between user components and the rest of the framework, hiding in this way system internals and allowing a rapid development of new components.

MDSplus is a framework for data acquisition and management widely used in the fusion community. It provides a common data interface to all the data types dealt with in a scientific experiment. The data abstraction is pushed further by the concept of expression. All data, ranging from simple scalars to whole python programs are represented in MDSplus expressions that are evaluated on the fly when required. Evaluating an expression means returning its value for scalars and arrays, or carrying out the computation specified in the expression itself. It is possible in this way to express in a natural way the data dependencies which characterize the possible complex structure of scientific experiments. MDSplus provides also support for high speed streaming data acquisition in distributed, multi reader and multi writer environments, and includes several support tools and Application Programming Interfaces (API) to data in several languages.

The functionality of MDSplus and MARTe has very little overlap, but their union covers most requirements for integrated real-time control and data acquisition. As a consequence, the two frameworks can be integrated in an effective way since:

– The lack of overlapped functionalities helps in the definition of an unambiguous definition of what the integration components have to merge.
– It is not necessary to implement new major components from scratch, and the integration mostly consists in providing wrappers to make the components from the different frameworks compatible each other.

The integration addresses two main features: configuration and data storage. Configuration in MARTe consists in the definition of a possibly large set of parameters required both by the core and the user components. In its default configuration, MARTe parameters are read during system startup from a configuration file, and they can be changed on the fly during offline (i.e. non real-time) operation. The MDSplus experiment model represents the most natural place for storing MARTe configuration data, in addition to the other experiment parameters. Data storage is required during real-time execution in order to save the streams of data handled by MARTe, both coming from sensors and produced by online computation.

A peculiar aspect of both MARTe and MDSplus is their highly modular approach that allows the installation of only those components that are required for the specific requirements. This eased their integration, mainly consisting in assembling components from both systems with a minimal development effort for interface adapter components.

3. MARTe-MDSplus integration

MDSplus pulse files are hierarchically organized and therefore sets of related data items can be grouped into subtrees in the main experiment tree. This represents the usual way data are organized in MDSplus. Pushing this concept further, groups of data items describing specific hardware devices can be grouped into subtrees. Different instances in the experiment of a given piece of hardware such as an ADC device are reflected in the experiment model and pulse file (the pulse is an experiment model filled with data acquired in a given pulse discharge) by different subtrees with the same structure. This approach has a strong similarity with Object Oriented programming, where device types represent the classes (i.e. describe the internal data structure) which can be instantiated during program execution. In general, MDSplus devices can be used whenever related data may appear in different instances, and this is the case for the MARTe components. For example, a MARTe module for PID control may be defined in several instances into the MARTe configuration, and each instance may require individual configuration parameters. The natural choice is therefore to define MDSplus devices to contain the configuration of the corresponding components in MARTe, originally defined in MARTe in a configuration file.

Please cite this article in press as: G. Manduchi, et al., Fast development of real-time applications using MDSplus and MARTe frameworks, Fusion Eng. Des. (2016), http://dx.doi.org/10.1016/j.fusengdes.2016.04.014