EPICS as a MARTe Configuration Environment

Daniel F. Valcárcel, Antonio Barbalace, André Neto, André S. Duarte, Diogo Alves, Bernardo B. Carvalho, Pedro J. Carvalho, Jorge Sousa, Horácio Fernandes, Bruno Gonçalves, Filippo Sartori, and Gabriele Manduchi

***Abstract*—The Multithreaded Application Real-Time executor (MARTe) software provides an environment for the hard real-time execution of codes while leveraging a standardized algorithm development process. The Experimental Physics and Industrial Control System (EPICS) software allows the deployment and remote monitoring of networked control systems. Channel Access (CA) is the protocol that enables the communication between EPICS distributed components. It allows to set and monitor process variables across the network belonging to different systems. The COntrol and Data Acquisition and Communication (CODAC) system for the ITER Tokamak will be EPICS based and will be used to monitor and live configure the plant controllers. The reconfiguration capability in a hard real-time system requires strict latencies from the request to the actuation and it is a key element in the design of the distributed control algorithm. Presently, MARTe and its objects are configured using a well-defined structured language. After each configuration, all objects are destroyed and the system rebuilt, following the strong hard real-time rule that a real-time system in online mode must behave in a strictly deterministic fashion. This paper presents the design and considerations to use MARTe as a plant controller and enable it to be EPICS monitorable and configurable without disturbing the execution at any time, in particular during a plasma discharge. The solutions designed for this will be presented and discussed.**

***Index Terms*—Monitoring systems, middleware, real-time systems.**

I. Introduction

THE Multithreaded Application Real-Time executor (MARTe) is a framework that allows the execution of the same code in different operating systems [1]. MARTe was built on top of the C++ BaseLib2 library which contains optimized objects for real-time applications. In particular it is used in real-time applications running in operating systems such as Linux with isolated central processing unit (CPU) cores, Linux-RTAI and VxWorks. These real-time applications have been related with, but are not limited to, nuclear fusion experiments [2]–[4].

The MARTe workflow for the application developer is to design and implement modules called generic application modules (GAMs). These execute sequentially in real-time threads and communicate by exchanging signals. A particular MARTe application can have several threads, each running its own set of GAMs. When designed as independent and sufficiently generic modules, they are also reusable on different applications.

Currently, there are several ways to interface with MARTe. Each GAM can have a dedicated HTML webpage to interface with the outside, useful to display information and/or to receive data from outside. There is also CODASlib, a module that integrates with the JET control and data acquisition system, where the messaging system allows to send a message to each running module, either from inside MARTe or from the outside using TCP/IP.

MARTe is configured using a well-defined structured language by means of a configuration file. When it receives the file, all the GAMs are destroyed, recreated, and reconfigured. This means stopping the system for enough time as to rebuild everything. At present, there is no standard way for introspection or atomic changes on the running modules without stopping the system.

The Experimental Physics and Industrial Control System (EPICS) is a set of Open Source software tools, libraries, and applications developed collaboratively and used worldwide to create distributed soft real-time control systems for scientific instruments [5]. In particular, it provides a rich set of tools for monitoring and configuring real-time systems, and it is currently used in several physics applications [6]–[9].

The EPICS Channel Access (CA) layer enables communication between subsystems. Each subsystem can be a CA server (CAS) or a CA client (CAC). A CAS contains internal data called process variables (PVs) and makes it available throughout the network so that every CAC can access it. An Input–Output Controller (IOC) is a software that can be simultaneously CAS and CAC, having also internal code that runs continuously. IOC is the EPICS standard way of implementing distributed controllers.

The ITER Tokamak is an experimental fusion device currently being built in France. It is a large physics experiment where a plasma has to be sustained from several minutes up to one hour [10]. These long discharges need sophisticated control systems that can be configured during the experimental sessions without disturbing performance.

ITER will have several fast plant system controllers (FPSC) which are elements of a distributed control system. According to [11], each provides local data acquisition, control, monitoring,

Manuscript received June 11, 2010; revised March 01, 2011, April 11, 2011, and April 17, 2011; accepted April 20, 2011. Date of publication June 02, 2011; date of current version August 17, 2011. This work has been carried out in the frame of the Contract of Association between the European Atomic Energy Community and Instituto Superior Técnico (IST) and of the Contract of Associated Laboratory between Fundação para a Ciência e Tecnologia (FCT) and IST. The content of the publication is the sole responsibility of the authors and it does not necessarily represent the views of the Commission of the European Union or FCT or their services.

D. F. Valcárcel, A. Neto, A. S. Duarte, D. Alves, B. B. Carvalho, P. J. Carvalho, J. Sousa, H. Fernandes, and B. Gonçalves are with Associação EURATOM/IST, Instituto de Plasmas e Fusão Nuclear, Laboratório Associado, Instituto Superior Técnico, P-1049-001 Lisboa, Portugal (e-mail: daniel.valcarcel@ipfn.ist.utl.pt).

A. Barbalace is with the EURATOM-ENEA Association, Consorzio RFX, 35127 Padova, Italy.

F. Sartori is with Fusion for Energy, 08019 Barcelona, Spain.

G. Manduchi is with Istituto Gas Ionizzati del CNR, Consorzio RFX - Associazione EURATOM/ENEA sulla Fusione, 35127 Padova, Italy.

Digital Object Identifier 10.1109/TNS.2011.2147799

TABLE I
CONFIGURATION PROTOCOL MESSAGES

	Parameters	Possible Outcomes
GetPropertyList	None	List of properties available
GetPropertyInformation	Property name	Property information (type, readable, writable, etc.) Property does not exist
GetProperty	Property name	Property value Property does not exist
SetProperty	Property name Property value	Property set OK Property does not exist Property not writable

```
Command = SetProperty
CommandUID = 1154263
PropertyName = CurrentSetpoint
PropertyConfiguration =
{
        Value = 5000.0
}
```

Fig. 1. Example of a property configuration message to configure an atomic floating-point value.

alarm handling, logging, event handling, and data communication functions.

All FPSC share data and communicate with the ITER Control, Data Acquisition and Communications (CODAC) system by the EPICS channel access layer. The interface with each FPSC internal data is done by means of defining properties [12], which can be read and written by the CODAC. Properties can be configuration properties or state properties. Configuration properties represent the inputs for the plant system and can be changed by CODAC. State properties reflect the set of internal values that are updated during real-time operation, and thus are read-only.

MARTe, as a real-time system, is an excellent candidate to be used as an FPSC. In order to do that, MARTe must be able to integrate with the EPICS platform.

There are two main issues to be addressed: the access to MARTe internal data and the integration on the CA layer of EPICS. The first comprises two parts: the access to real-time data transferred between GAMs (signals) and the atomic configuration of MARTe modules. The latter regards how MARTe makes the data available to EPICS as PVs, handling issues such as alarms and other EPICS specific details.

This paper discusses the approaches designed to address the integration of MARTe with the EPICS platform and enable it as a configurator for MARTe.

II. INTERFACING WITH MARTE

A. Configuration Protocol and Library

Similarly to the definition of the FPSC properties, in the context of this work, a property is a named block of data belonging to a MARTe module that can be monitored and/or changed, reflecting the internal state of the system or configuration inputs.

The proposed interface method to MARTe's modules is based on the exchange of messages between an entity and the module. This scheme conforms to a defined protocol that states the messages the module must be able to handle and the replies the entity will receive in response. Table I illustrates the messages defined in the protocol and the responses that may be generated. Fig. 1 shows an example of a property configuration message sent to a MARTe module.

This protocol is implemented as a C++ library and allows developers to enable their MARTe modules for configuration with the protocol.

With this library, many types of data can be accessed. These are atomic scalar values (e.g., int or float), atomic array values (e.g., int* or float*), or nonatomic values such as BaseLib2 named objects.

B. Signal Access

To access the MARTe internal signals and translate them into read-only EPICS PVs, there are two possibilities, depending on whether the access is internal or external to MARTe. The access external to MARTe may be accomplished by data streaming [13]. On the other hand, if it is an internal access, then the signals are accessed directly. Signals are stored and handled by the dynamic data buffer (DDB) structure, which stores them in memory in a highly efficient manner.

III. INTERFACING WITH EPICS

A. Traditional IOC Interface

A standard IOC can be used for the interface although it has to be customized for the intended application (Fig. 2). This IOC must handle EPICS specific issues, such as handling permissions or alarms, and map properties and signals from MARTe into EPICS PVs. Usually this would be done by implementing one or more EPICS records that would access to all or a subset of the exported properties and signals. It has to be stressed that at the current EPICS version (v3.14.10) the record definition is static, meaning that, to change the PV definition, the code has to be changed and recompiled. Thus, it is not possible to develop a generic IOC to access MARTe, but rather the IOC will be specific for a particular application.

The configuration of properties can be handled by the configuration library (Section II-A). To send the configuration messages to the internal module that is to be configured, the messages have to be sent via the MessageBroker module. This module runs inside MARTe receiving the messages and

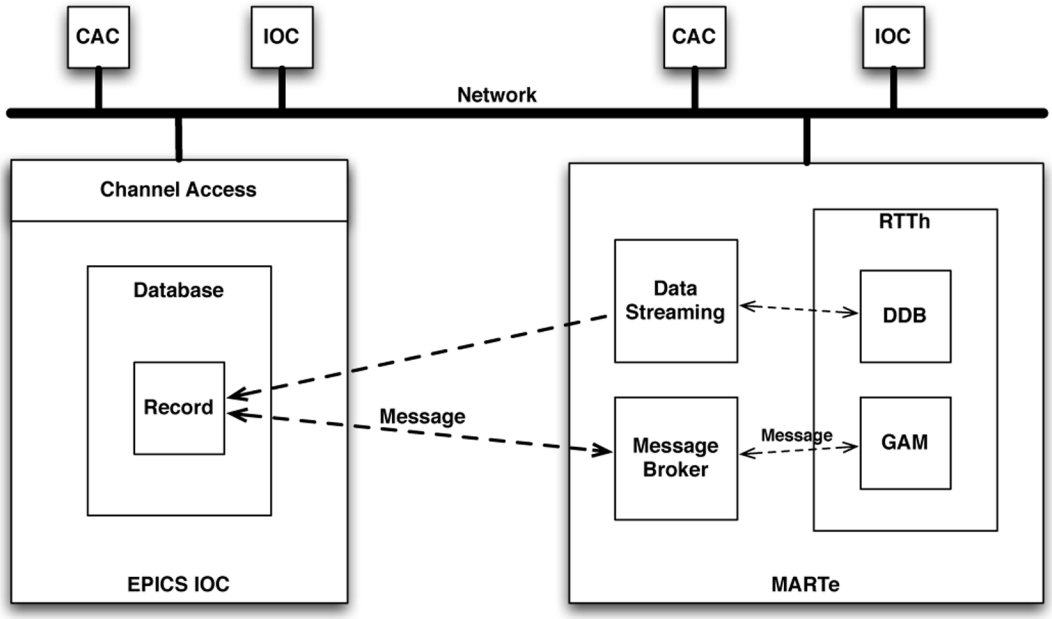

Fig. 2. A traditional IOC to bridge EPICS accesses to MARTe. RTTh: real-time thread.

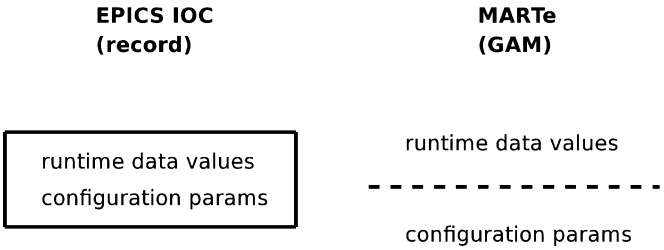

Fig. 3. Data separation concept between EPICS and MARTe.

sending the replies through TCP/IP connections. The translation of MARTe internal signals into PVs can be done collecting data via data streaming.

B. MARTe–IOC Interface

A traditional IOC can be replaced by a MARTe-based one. For this, MARTe must implement the Channel Access protocol, supporting the application-specific devices and offer, at the algorithmic level, the same computational blocks that a developer can find in a traditional IOC. This is possible given the similarities between the two systems.

EPICS' computational blocks are called records, whereas MARTe's are called GAMs. The basic difference between records and GAMs lays in how the access to data values and configuration parameters is structured, as in Fig. 3. EPICS allows the user to access runtime values and configuration parameters in the same way; i.e., EPICS does not make any distinction between these kind of data. MARTe typically separates runtime data access to offline component configuration. MARTe's approach relies on the dictates of real-time operating systems theory: every nondeterministic procedure must be done during the initialization phase (or better, during offline). This difference is enforced by the subroutines (EPICS IOC) and methods (MARTe) of the different components.

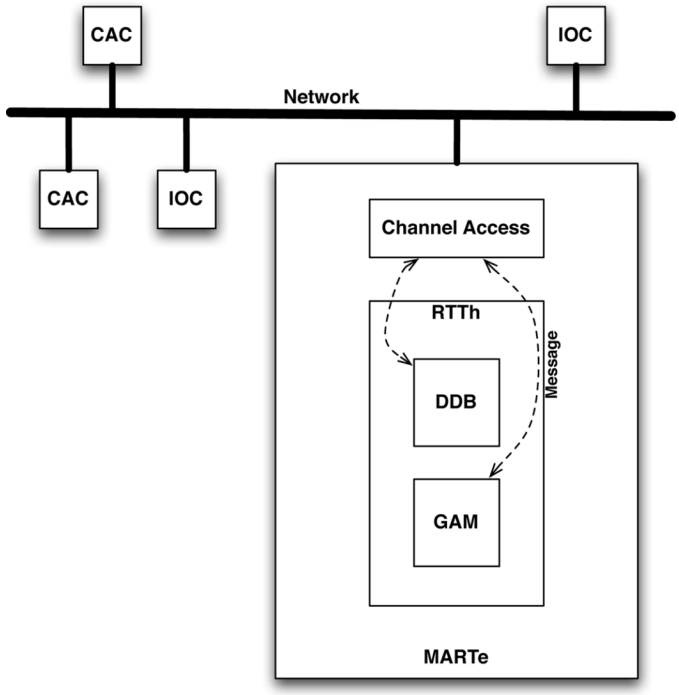

Fig. 4. Using MARTe as an IOC. RTTh: real-time thread.

Fig. 4 depicts the MARTe IOC integrating on the CA layer of EPICS. Internally it can access properties via the configuration library and signals directly from the DDB.

C. GAM Approach

The approaches to integrate MARTe properties and signals as EPICS PVs just described do not solve the reverse question—the access to EPICS PVs from inside MARTe. Considering that PVs are usually used as part of algorithmic calculations, the approach that makes more sense to solve this issue

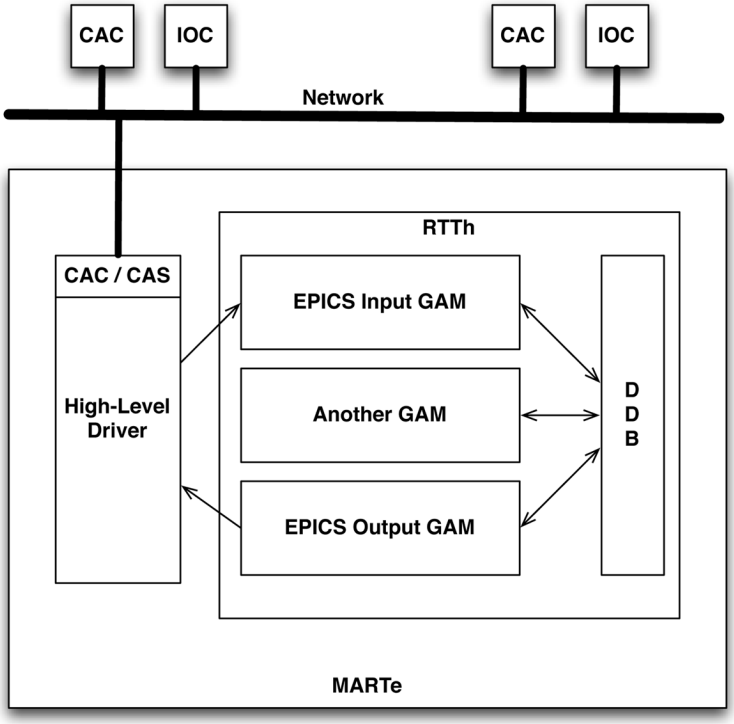

Fig. 5. EPICS GAMs inside a MARTe real-time thread to access EPICS process variables. The EPICS Input GAM stores read PVs on the DDB, any GAM may access the values, and the EPICS Output GAM translates signals from the DDB into PVs.

is depicted in Fig. 5. It consists in using a high-level driver that connects with the CA layer, an input GAM (EPICS Input GAM) that reads a set of defined PVs from the driver, and an output GAM (EPICS Output GAM) that handles the write back of the (eventually processed) signals to the driver along with any limits or alarms. Leaving the CA connection up to a high-level driver has the advantage that it can run in an isolated CPU core and avoid disturbing the real-time performance of the system.

This GAM approach also makes it possible to use the EPICS Input GAM as an EPICS controlled signal generator. Besides accessing external PVs, acting as a CAC, it can be used to create new PVs acting as a CAS. These PVs can be changed by external CACs or IOCs, and the changes reflect in the exported MARTe signals.

IV. Discussion

EPICS is a well-established framework for distributed systems. In particular, it provides a flexible configuration environment for its subsystems. For example, the Control System Studio (CSS) software [14] provides a standard monitoring and configuration environment for PVs. This means MARTe can benefit from this simply by integrating on the CA layer. All aspects addressed in the described integration approaches have their strong points and drawbacks, and it is important to underline these in order to decide the best approach, or combination of approaches, for a particular application.

The configuration protocol/library enables the configuration of properties without major changes in MARTe. Only the modules that intend to take advantage of enabling monitoring/configuration at any time require code changes/recompilation. The nonenabled modules simply ignore the configuration messages they might receive. The fact that the configuration is message based implies that external configurations are possible through TCP/IP. The drawback of message-based configuration is that the message decoding can take more time than if the approach was done by calling a direct method in the module.

For the integration with EPICS, three approaches were described: the traditional (external) IOC, the MARTe IOC, and the GAM approach.

A traditional IOC allows us to integrate MARTe in the EPICS layer without any new module development for MARTe. Also it can be used on a different computer than the one where MARTe runs. On the other hand, it has the drawback that the approach is static, and thus an IOC needs to be developed according to the application at hand.

The MARTe IOC approach is dynamic as it queries the DDB for the signals and the modules for properties through the configuration library. It runs on the same computer where MARTe runs, and, depending on the application, this can be considered a drawback.

The GAM approach solves the issue of importing PVs into MARTe and creating signals that are not just read-only. What can be considered a drawback in this approach is that it handles just signals and not properties; it is necessary to use another approach to handle properties.

The traditional IOC approach, along with the configuration library, have been tested successfully in [15], and the MARTe IOC approach is currently under active development.

References

[1] A. Neto *et al.*, "MARTe: A multi-platform real-time framework," *IEEE Trans. Nucl. Sci.*, vol. 57, no. 2, pp. 479–486, Apr. 2010.

[2] F. Sartori *et al.*, "The PCU JET plasma vertical stabilisation control system," *Fusion Eng. Design*, vol. 85, no. 3–4, pp. 438–442, Jul. 2010.
[3] D. F. Valcárcel *et al.*, "Real-time software for the COMPASS Tokamak plasma control," *Fusion Eng. Design*, vol. 85, no. 3–4, pp. 470–473, Jul. 2010.
[4] P. J. Carvalho *et al.*, "ISTTOK plasma control with the tomography diagnostic," *Fusion Eng. Design*, vol. 85, no. 2, pp. 266–271, Apr. 2010.
[5] Argonne National Laboratory, Experimental Physics and Industrial Control System [Online]. Available: http://www.aps.anl.gov/epics
[6] Y. Yan *et al.*, "EPICS interface to Libera electron beam position monitor," *Nucl. Sci. Techniques*, vol. 19, no. 2, pp. 65–69, Apr. 2008.
[7] K. H. Kim *et al.*, "The KSTAR integrated control system based on EPICS," *Fusion Eng. Design*, vol. 81, no. 15–17, pp. 1829–1833, Jul. 2006.
[8] O. A. Makarov *et al.*, "EPICS controlled sample mounting robots at the GM/CA CAT," *Nucl. Instrum. Methods Phys. Res. A, Accel.. Spectrom. Detect. Assoc. Equip.*, vol. 582, no. 1, pp. 156–158, Nov. 2007.
[9] W. A. Watson, III *et al.*, "The CEBAF accelerator control system: Migrating from a TACL to an EPICS based system," *Nucl. Instrum. Methods Phys. Res. A, Accel.. Spectrom. Detect. Assoc. Equip.*, vol. 352, no. 1–2, pp. 118–121, Dec. 1994.
[10] R. Aymar, P. Barabaschi, and Y. Shimomura, "The ITER design," *Plasma Phys. Controlled Fusion*, vol. 44, no. 5, p. 519, 2002.
[11] *"Plant Control Design Handbook,"* ver. 5.2, ITER CODAC Section, Feb. 2010.
[12] F. Di Maio, The CODAC–Plant System Interface, ver. 1.1, Dec. 2009, IDM Number ITER_D_34V362.
[13] L. Zabeo *et al.*, "Continuous data recording on fast real-time systems," *Fusion Eng. Design*, vol. 85, no. 3–4, pp. 374–377, Jul. 2010.
[14] DESY, Control System Studio [Online]. Available: http://css.desy.de
[15] B. Gonçalves *et al.*, "ITER prototype fast plant system controller," presented at the 26h Symp. Fusion Technol., Oporto, Portugal, Sep. 27–Oct. 1, 2010.