MARTe in JET April 20, 2010

April 20, 2010 - EFDA Feedback Control Working Group Meeting

D. Alves[1] G. De Tommasi[2] A. Neto[1] F. Sartori[3] and EFDA-JET PPCC contributors

[1]Associação EURATOM/IST, Instituto de Plasmas e Fusão Nuclear
[2]EURATOM-ENEA-CREATE, Università di Napoli Federico II
[3]Fusion For Energy

Outline

Where we started from - mid 90s

From JETRT to MARTe

The eXtreme Shape Controller

MARTe in JET

Additional modules of the XSC

The new Vertical Stabilization System - VS5

EFCC Voltage Amplifier Controller

PPCC Control systems in the mid 90s

PPCC systems for plasma magnetic control

The two main systems run at JET by the Plasma Position and Current Control Group were (and still are!):

Shape Controller (SC) C code deployed on a VxWorks/VME/Motorola68k platform

the Vertical Stabilization System (VS) C code deployed on 4 Texas Instruments DSPs

- The code was *tailored* for the specific platform
- Lack of modularity
- Different software solutions to interface with the JET software infrastructure (pre-pulse system configuration, post-pulse data collection,...)

F. Sartori et al.,
The Joint European Torus - Plasma position and shape control in the world's largest tokamak,
IEEE Control Systems Magazine, vol. 26(2), Apr. 206

A new framework for RT applications

Motivation

- In 2001/2002 the revamping of the SC was planned in order to add the eXtreme Shape Controller algorithm (XSC)
- Within the PPCC group, it was decided to move to a common framework for the development of real-time application

Aims

- Standardize the development of real-time application
- Increase the code reusability
- Give the possibility to separate the user application from the software needed to interface with the plant infrastructure
- Reduce the time needed for commissioning.

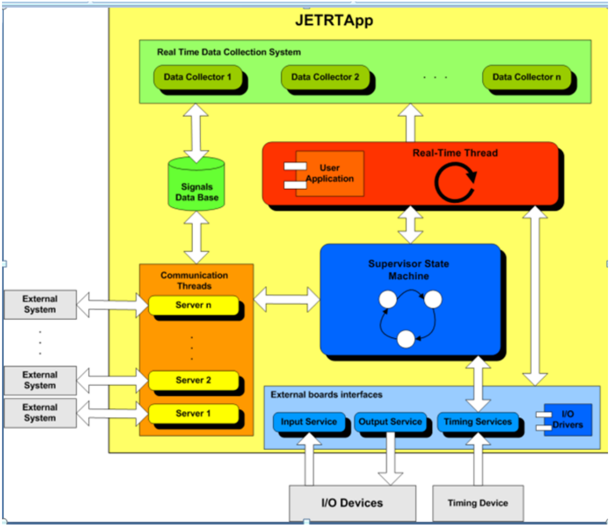
Requirements

The new framework would have been:

- portable (multi-OS and multi-platform)
- modular – the user application would have been easily *plugged* into an executor of real-time application
- written in C++ (at that time C++ was not a CODAS *standard*)

JETRT

- The JETRT framework was developed in 2002/2003 to deploy the XSC
- JETRT is based on the cross-platform BaseLib library (developed within the PPCC group)



Separation between application and infrastructure software

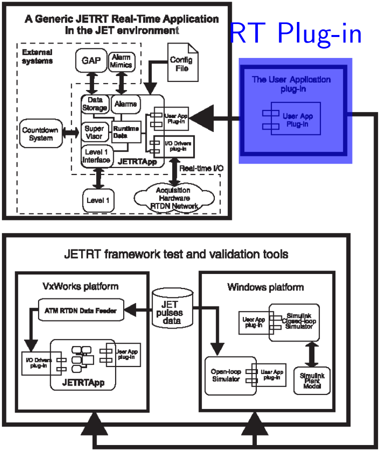
Why we want to separate application from infrastructure software?

- Scientist (process experts) can abstract from the plant interfaces
- Increase code reusability
- Achieve standardization

Real-time application plug-in

As a result we have a **Real-time Application Plug-in** that can be used to:

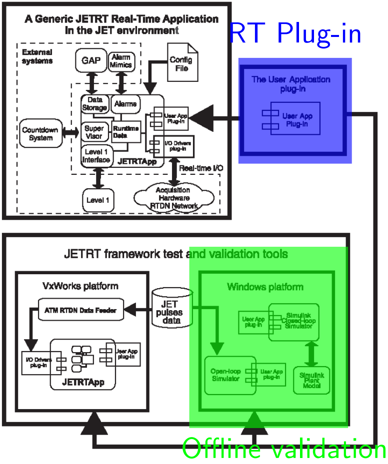
-
-
-



Real-time application plug-in

As a result we have a **Real-time Application Plug-in** that can be used to:

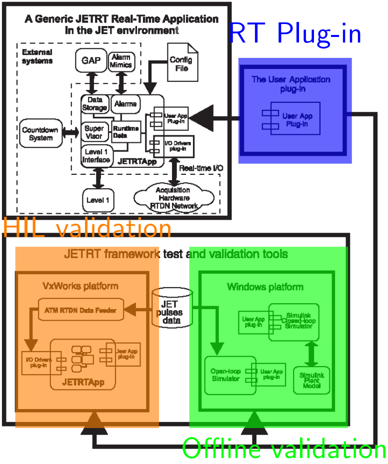
- perform offline validation against a plat model
-
-



Real-time application plug-in

As a result we have a **Real-time Application Plug-in** that can be used to:

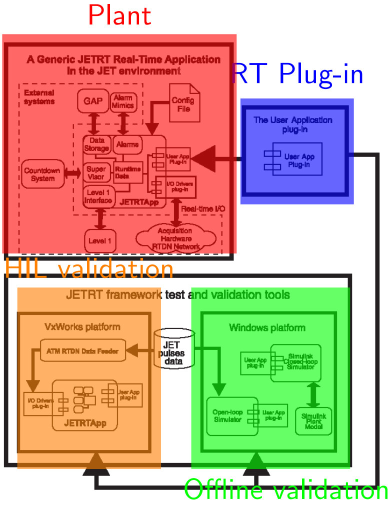
- perform offline validation against a plat model
- perform real-time validation with hardware-in-the-loop
-



Real-time application plug-in

As a result we have a **Real-time Application Plug-in** that can be used to:

- perform offline validation against a plat model
- perform real-time validation with hardware-in-the-loop
- run the real-time system on the plant



XSC

- The new SC (including the XSC) has been deployed on a 400 MHz G4 PowerPC running VxWorks
- 2 ms control loop (but it can easily run at 1 ms)

Commissioning of the JETRT framework and of the XSC

- Thanks to portability, an exhaustive debug of both the JETRT framework and the XSC was performed offline on a Windows–based platform
- Only 3 days of testing on the plant were needed to commission the new system

XSC

JETRT limitations

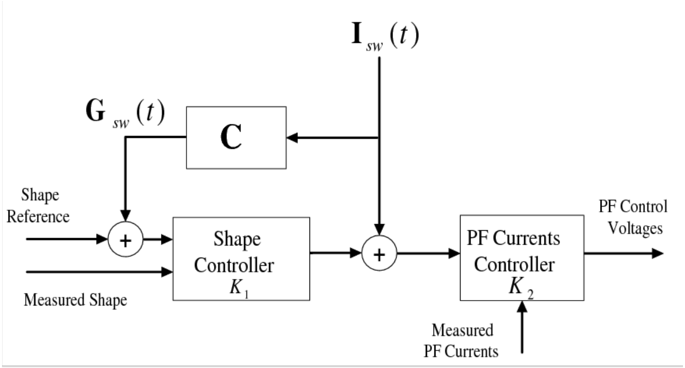
Although it was a first attempt, JETRT didn't provide a real separation between the user application from the plant-interface software!

From JETRT to MARTe

- *More* modularity → Generic Application Modules (GAMs)
- *Real* separation → Dynamic Data Buffer

Sweeping for the XSC

The first version of GAMs has been adopted in 2007 to implement the strike–points sweeping algorithm for the XSC



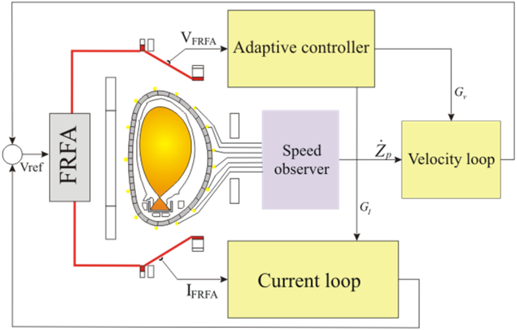
MARTe–based system at JET

Since 2008 several MARTe–based systems have been deployed at JET:

- betap–li ?? (??) please Filippo help us
- walls ?? (??) Filippo help us
- new VS system (2008/2009)
- new EFCC controller (2009)

VS - Introduction

- Elongated tokamak plasmas are susceptible to a vertical axisymmetric instability
- Dedicated VS system required
- Essential system for operation
- Growth rate of 1000 s^{-1}
- Loss of control can produce forces in the order of the 100's of tonnes



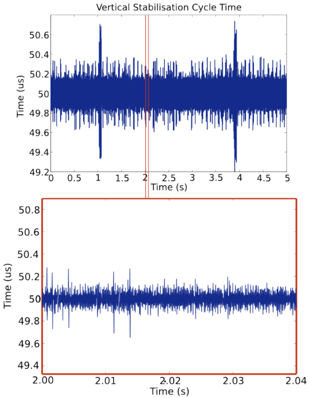
VS - New Hardware

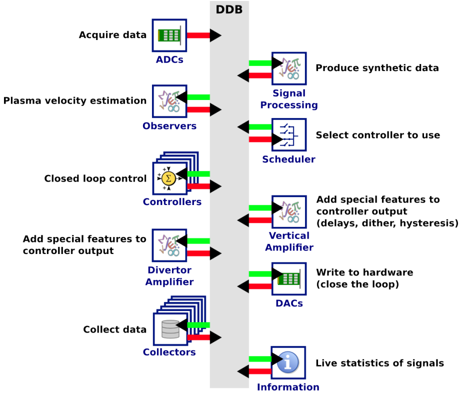
- Elongated tokamak plasmas are susceptible to a vertical axisymmetric instability
- 192 signals acquired by ADCs and transferred at each cycle
- 50 μs control loop cycle time with jitter [illegible]s
- Always in real-time (24 hours per day)
  - 1.728×10^9 50 μs cycles/day
  - Crucial for ITER very long pulses

VS Requirements

- 18 GAM instances
  - Altogether execute in less than 40 μs
  - Synchronization always achieved within 0.8 μs
- 192 signals acquired by ADCs and transferred at each loop
- Enable advanced experimental features
  - ELM pacing
  - Complex time windows with different controller features and settings

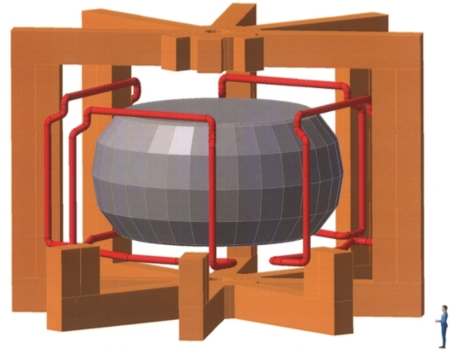
MARTe for the new VS



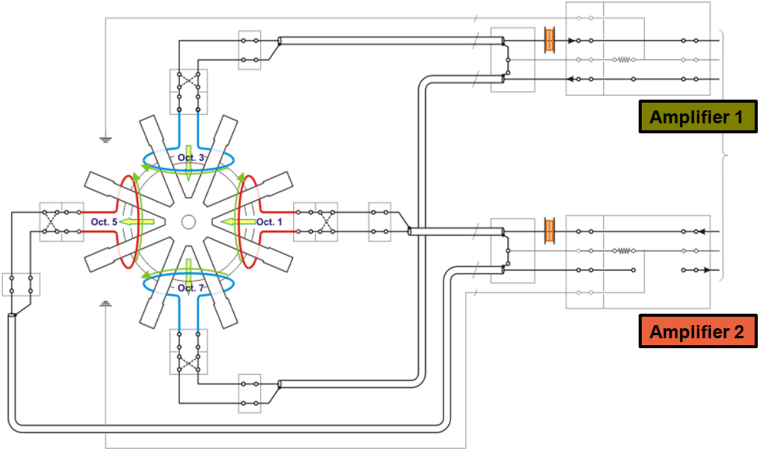


EFCC Voltage Controller - Introduction

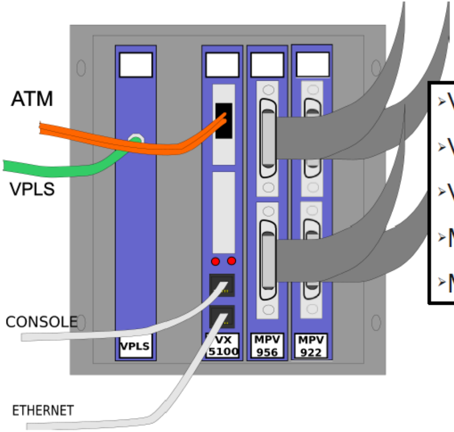
- EFCCs, what do they do?
  - ~~They~~ change magnetic field topology at the plasma boundary
- Why is it important?
  - Instability mitigation and ELM control
- How?
  - By controlling the current in the EFCCs we can control the magnetic field
- Who?
  - The session leader sets the required current waveforms



EFCC controller – Schematic

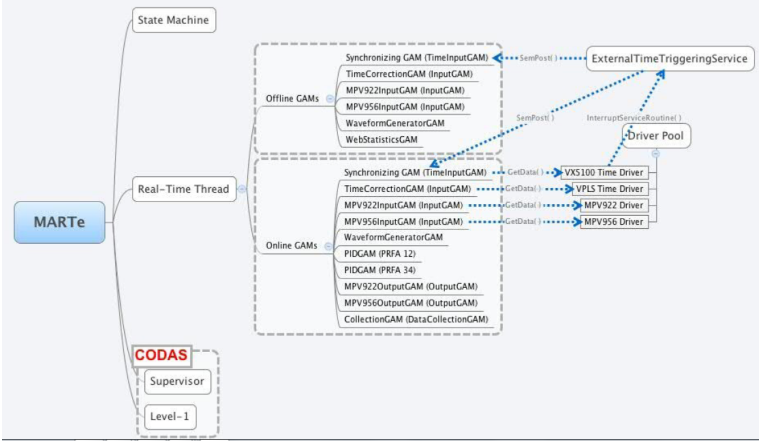


EFCC controller – Hardware

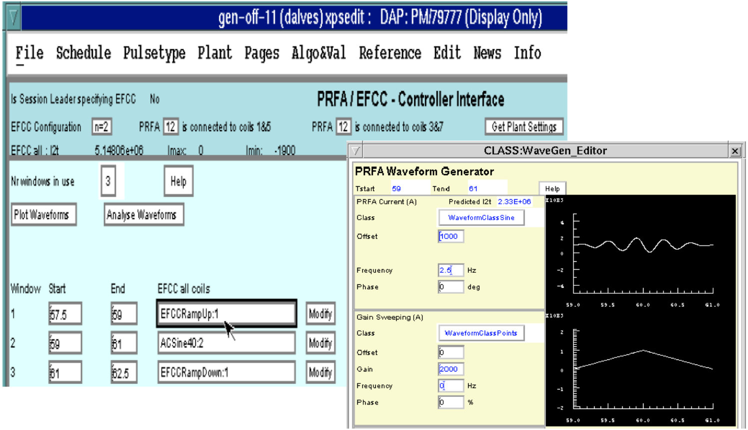


- VME-based technology
- VPLS (timing)
- VX5100 (cpu)
- MPV956 (analog I/O)
- MPV922 (digital I/O)

EFCC controller – Software



EFCC controller – Configuration



EFCC controller – Experimental results