25th IEEE Real Time Conference, Poster 214

May 25-29, 2026

A common framework for real-time structured data communication in fusion devices

N. Ferron[1], C. Galperti[2], G. Manduchi[1,3], A. Neto[4], A. Segovia[2], F. Felici[2]

[1] *Consorzio RFX (CNR, ENEA, INFN, Università di Padova, Acciaierie Venete SpA), 35127 Padova, Italy*
[2] *EPFL-SPC, 1015 Lausanne, Switzerland*
[3] *CNR-ISTP, 35127 Padova, Italy*
[4] *Fusion for Energy, 08019 Barcelona, Spain*

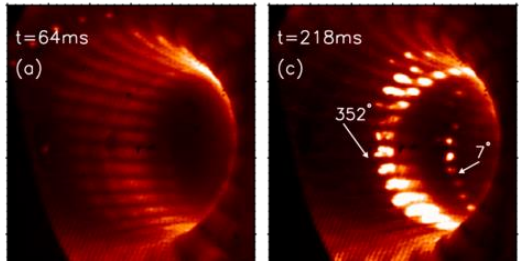


Fig. 1: RFX-mod, an RFP confinement device

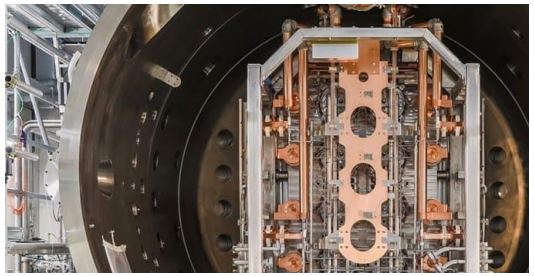


Fig. 2:SPIDER, the linear accelerator for ITER

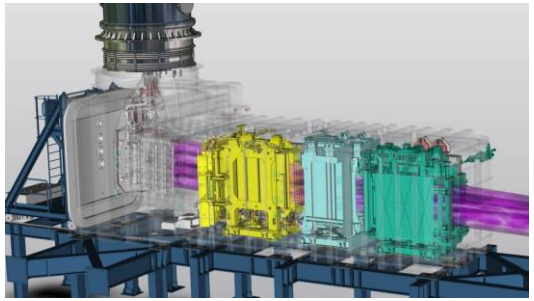
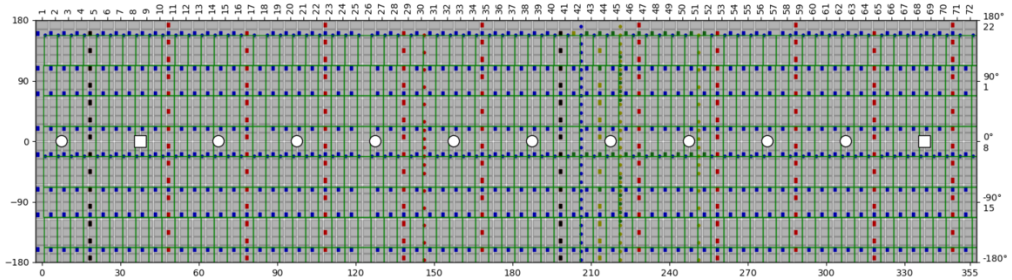


Fig. 3: MITICA, the 1:1 NBI prototype for ITER

1. At Consorzio RFX we host **three main experiments**:
    1. RFX-mod2
    2. SPIDER
    3. MITICA
2. To drive these experiments we use **MARTe2** + **MDSplus** + **MATLAB/Simulink**©
    a. MARTe2: C++ real-time framework
    b. MDSplus: fast data storage and management
    c. MATLAB/Simulink: algorithm design and simulation
3. This mix is/will be/was also used JET, TCV, COMPASS, SPARC, DTT [and ITER]
4. They were exchanging **only unstructured/plain** signals and parameters:



```
+IOGAM_MDSReader2 = {
  Class = MDSWriter
  InputSignals = {
   // ==================== triaxial pickups (poloidal) set 2 ====================
   BP_1102      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1102_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1106      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1106_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1110      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1110_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1114      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1114_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1118      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1118_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1122      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1122_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1126      = { Type = float64 NumberOfElements = 1 DataSource = MDSReader2 }
   BP_1126_Qual = { Type = uint8   NumberOfElements = 1 DataSource = MDSReader2 }
   // ... to be continued (this configuration files is 2600 lines long)
```

Allowing the work to flow

1. We develop a modular and flexible set of tools to provide **structured data communication** in a popular PCS configuration in fusion devices
2. With structures, the experimental data:
   a. remains tidy, easy to maintain and fast to consult
   b. is less error-prone
   c. implicitly carries additional information about each dataset
3. Structured data allows the workflow in fig. 4
   1. ML/S to MDSplus: custom APIs
   2. MDSplus to MARTe2: MARTe2 component
   3. Direct communication between ML/S and MARTe2 is also possible
4. **Conclusions**: this work
   1. bridges the architectural gaps between software
   2. establishes a unified framework from design to discharge
   3. enhances data integrity and metadata retention without sacrificing real-time performance
   4. provides a robust, scalable foundation essential for handling the increasing data complexity of next-generation fusion devices

```
+Types = {
  Class = ReferenceContainer
  +Triaxi_Signals_t = {
    Class = IntrospectionStructure
    BP_Signal = { NumberOfElements = {1} Type = float64 }
    BP_Qual   = { NumberOfElements = {1} Type = uint8   }
  }
}
+IOGAM_MDSReader2 = {
  Class = IOGAM
  InputSignals = {
    // ==================== triaxial pickups (poloidal) set 2 ======================
    BP_Signals_Struct = { Type = Triax_Signals_t NumberOfElements = 72 DataSource = MDSReader2 }
```

Fig. 4: the same configuration now takes 1 line!

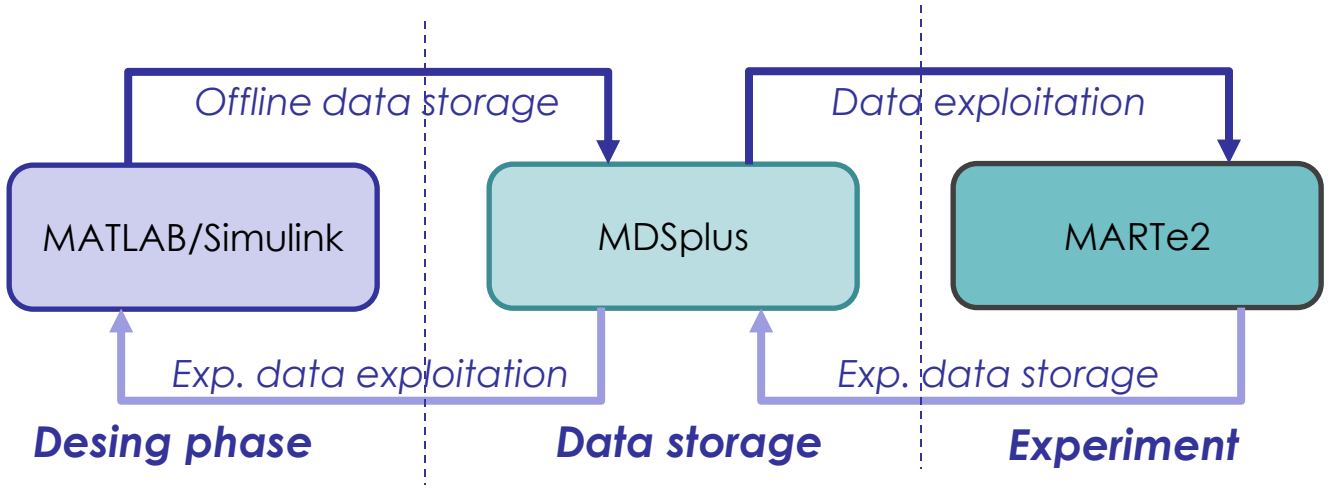


Fig. 5: Workflow allowed by structured data