Outlook on software framework in low and intermediate energy nuclear physics

Adrien Matta
Normandie Univ, ENSICAEN, UNICAEN, CNRS/IN2P3, LPC Caen, 14000 Caen, France
E-mail: matta@lpccaen.in2p3.fr

Abstract. Over the past decades nuclear physics experiments have seen a drastic increase in complexity. With the arrival of second generation radioactive ion beam facilities all over the world, the run for exploring more and more exotic nuclei is raging. The low intensity of RI-beams requires more complex setup, larger solid angle coverage and detection of a wider variety of charged and neutral particles. Design, construction and operation of a variety of complex instruments used in such experiments require many software developments. The short lifetime of experimental setups and multiple combinations of instruments demand a strong methodology. As the community is shifting to this new paradigm, the quest for the optimum framework is becoming central in the field. In this outlook we will introduce the specificity of the nuclear physics community, technical needs of such frameworks, and give an overview of the existing ones, with an emphasize on the difficult balance between computing performances, versatility and integration with other frameworks.

1. Introduction

Nuclear physics is the quest for answering three complementary questions: the question of structure, or the quest for a universal model describing the properties of known nuclei, and predicting the properties of unknown one, the question of dynamic, or what happens when two nuclei collide at a given energy and lastly, the Equation Of State (EOS) of finite and infinite nuclear matter. Those questions span four order of magnitude in energy and experimental studies are carried out all over the world as depicted in Figure 1.

Over the past decades low energy, from 0 to 100 MeV, and intermediate energy, from 100 to 1000 MeV, nuclear physics experiments have seen a drastic increase in complexity. In the early 2000s, the community started preparation for a new generation of RI-beam installations, such as HIE-ISOLDE, SPIRAL2, FAIR [1], RIBF [2], FRIB [3], and ISAC I/II - ARIEL [4]. Such installations produce much more exotic RI-beams, but also at lower beam intensities. This led to the development of more complex detection systems and their combinations to make the most of available beams. The community started a more extensive use of the GEANT4 [5] and ROOT [6] toolkit to develop simulation and analysis of such detectors and helps with their designs.

As the community is shifting to this new paradigm, the quest for the optimum framework is becoming central in the field. In this outlook, I will introduce the specificity of the nuclear physics community, technical needs of such framework, and give an overview of the existing one, with an emphasize on the difficult balance between computing performances, versatility and integration with other frameworks.

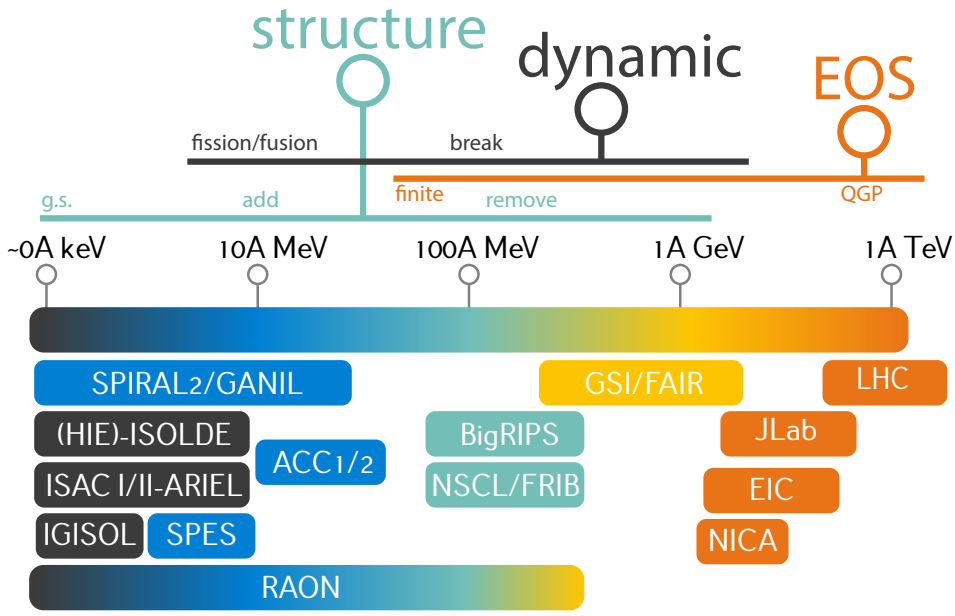

Figure 1. The field of nuclear physics is spanning various thematics, a large energy range and many facilities around the world. This leads to different needs by different sub-communities.

2. Framework

The use of software frameworks allows for more efficient development by providing both virtual interfaces and concrete implementation that can be reused selectively throughout the software under development. A well thought framework therefore allows *(i)* quick addition of new functionalities (algorithm, data i/o, detector geometry,...) and *(ii)* dynamically calls those functionalities without modification of the framework.

Another aspect specific to scientific software development is readability of the code. A well structured framework allows the emergence of well defined pattern that helps the user to find a specific implementation easily within the source code. On this aspect a strong contrast exists between industry, where the final product running performances is the priority, and science, where the final product is the algorithm itself rather than the executable. Indeed, results from any modern experiments are coming from a series of data manipulation and software associated to this manipulation is therefore an integral part of the results. Maintenance and publication of software associated with scientific results are then necessities to make the results meaningful.

Until the mid 2000s, low to intermediate energy nuclear physics were generally of small scale, with fewer electronic channels and detector systems. No strong needs for a proper methodology associated with the analysis software development was needed. In most instances, a specific, non-modular analysis program was developed specifically to produce the results associated with a data set.

With more complex and long term developments in mind, the need for more structured workflows arose, and the use of software frameworks appeared. Three frameworks took traction within the community, with different approaches to the task ahead. As early as 2002, Kaliveda [7], dedicated to the Indra/Fazia collaboration started. In 2006, FairRoot [8, 9, 10], based on earlier works on a virtual Monte-Carlo interface was initiated. In 2008, the development of *nptool* [11, 12] started with a focus on small scale, short lived, experiments.

3. Kaliveda

Kaliveda is the pioneer in the field, starting as early as 2002. Indeed the Indra collaboration was at the time one of the largest collaborations at intermediate energy. The study of nuclear dynamic was done with a single detector system, brought on different facilities. This made Indra one of the first *travelling detector* system in our field.

Because of the complex analysis associated with multi-fragmentation events, it took several years to analyse a single data set. To overcome the issues a mechanism to properly reference data sets and calibrations was needed, as well as interfaces allowing easy comparison of competing algorithms.

On the one hand, those analysis require a careful evaluation of the detector response, but on the other hand, the detection of 50 to 100 MeV charged particles did not require complex physics to simulate. The package focuses on providing accurate energy losses of charged particles in this range of energy but ignores any other processes and is not able to simulate neutral particles such as neutron and γ-rays. This eliminates the need to interface Geant4, leaving the sole dependency to the ROOT toolkit.

Many complex algorithms able to automatically perform E-ΔE identification of charged particle were integrated in the software. More recently, the addition of the Fazia detector array expanded the capabilities of the framework to Pulse Shaped Analysis (PSA) in nTD silicon detectors [13].

4. FairRoot

The FairRoot [8, 9, 10] framework started around 2006 and is based on earlier development of a virtual Monte-Carlo (VMC) interface. The goal was to allow users to perform simulation with different toolkits, such as Geant4, Fluka [14, 15] and MCNP [16], within the same environment. This approach was essential for the development phase of large scale equation of state experiments such as CBM and PANDA [17, 18] planned to run at the upcoming FAIR facility.

After a first release of CbmRoot[19] in 2004, the PANDA collaboration joined the effort and the base common frameworks were aggregated within the generic FairRoot framework. Over the following decades, many similar projects started using FairRoot outside of the FAIR facility perimeter, such as SHIP (CERN) [20], or MDP (NICA) [21]. A notable case of FairRoot based software packages is EnsarRoot[22]. This package was aimed at providing a FairRoot based environment for low energy nuclear physics experiments.

The FairRoot framework focuses on providing the necessary services to access computing infrastructure to large scale experiments. It provides the tools to manage and access databases of relevant parameters, I/O of complex geometries, as well as event display. A notable development is the FairMQ layer, allowing transparent communication of different components of the software locally or through network. This development effectively allows easier deployment of distributed computing. These components are now used by the ALICE (CERN) collaborations through the AliceO2 package [23].

One of the difficulties in managing large projects such as FairRoot is the handling of dependencies. Relying on third party packages is a great way of improving performances while reducing development time, however a mechanism assuring users have all the necessary dependencies installed with the correct version is difficult. FairRoot developed the Spack based FairSoft package to take care of installing all the dependencies.

5. nptool

nptool [11, 12] development started in late 2008, with the goal of streamlining the analysis and simulation of low energy transfer experiments. Two axes of developments were followed and formed the core philosophy of the framework. First, an universal approach was taken from the

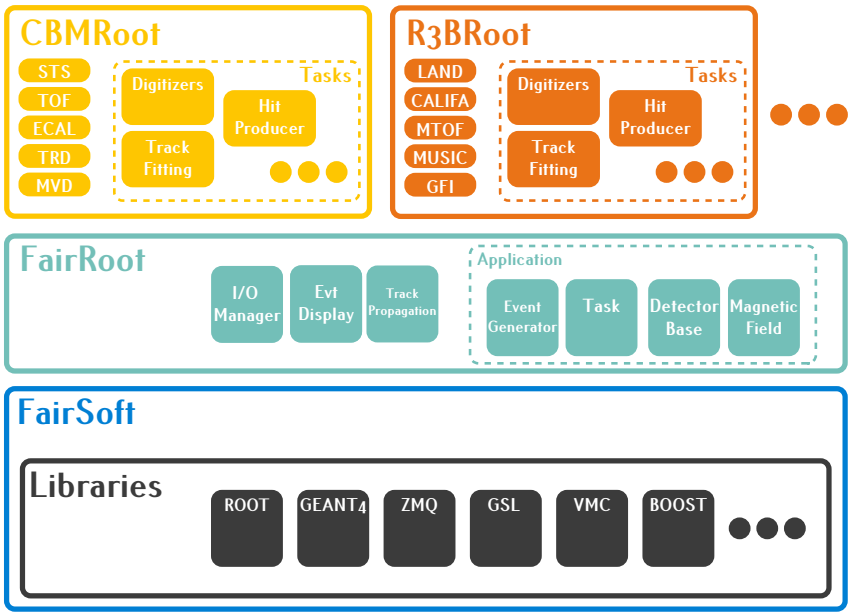

Figure 2. Overview of the different component of FairRoot based packages such as CMBRoot[19] and R3BRoot[24]. At least nine FairRoot based packages exist [19, 20, 21, 22, 24, 25, 26, 27, 28].

very start, making the framework both detector and physics agnostic. The second essential aspect of the design was to bring all the possibilities of both ROOT and Geant4 with minimum training for the end user.

The main difficulty in planning, performing and analysing low energy nuclear structure experiment is the fact that each setup is dedicated to specific physics case. The specific of the kinematic, the type of particle to detect and the beam characteristic make the use of a single static setup difficult. Modularity is therefore paramount at two levels: building of heterogeneous experimental setup, *i.e.* modular geometries, and assembling observable from all component in a meaningful results, *i.e.* modular analysis.

First, assembling heterogeneous experimental setup comprised of several modular sub-systems should be easily be done. To this end, we use a set of human readable input files that describe both the physics at play and the detector setup.

Secondly detectors are used to produce various type of observables depending on the experiment and the combination of the detectors employed. Separating the experiment invariant part of the analysis from the experiment specific one is therefore essential. By implementing a specific analysis project for each experiments the two get naturally separated while providing maximum flexibility to the end user.

Each detector is implemented as a group of classes that form a plugin library associated with an ASCII token. The token is used in the input file to trigger the loading of the library and the registration of the detector to a *Detector Factory*. The framework takes care of instantiating and initializing the required classes depending on the use case as presented in Figure 3.

Over time *nptool* started integrating many useful libraries to help physicists perform complex analyses and share developments time between collaborations. For instance, a full suit of classes is available for calibration of silicon detectors, and is used commonly by the MUST2 [29], SHARC [30], and TIARA [31] collaborations. Tools for track reconstruction using the RANSAC algorithm is shared between the MINOS TPC [32] and ACTAR TPC [33] analyses. Recently cross-talk rejection for neutron scintillator arrays was initiated and work to deliver

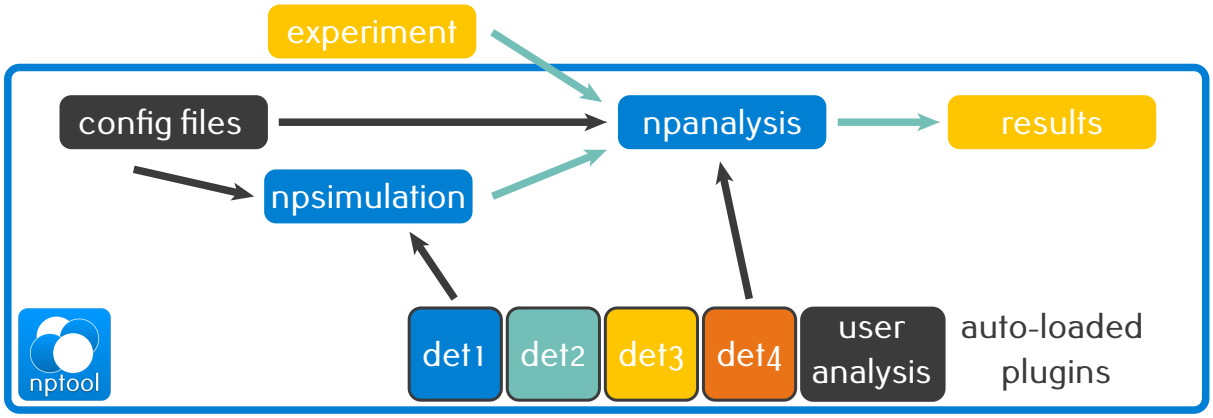

Figure 3. A simplified schematic of the work flow. The end user manipulates a handful of configuration files lets the framework deal with loading of the necessary libraries at run time.

a comprehensive library to the community is ongoing with collaborations between MSU, LPC Caen, and CEA-DAM.

With now more than 70 detector systems provided along the framework, maintenance of the package is proved to be more difficult. To this end a new version [34] is in preparation where all the plugins will be externalised. The framework will come with utilities to install, uninstall, and update plugins required by the end user. In addition the possibility to duplicate and modify a given plugin within the scope of a project will be provided.

This architecture will hopefully help tackle the upcoming challenge of Open Science for short lived experiments, helping the community to provide and publish self-contained, readable and achievable, software packages associated with any data set.

6. Conclusion

20 years of software development brought the community of low to intermediate energy nuclear physics to a point where mature, modular, and usable software frameworks are now widely available. All these developments will hopefully help tackle the challenges that represent open science in such a varied field of study. However, developing and deploying the tools, as well as educating different communities is a long road. The integration of more collaborations within those frameworks, as well as their inter-operability will play an essential role in bringing the community together to maximise the potential of all available data sets.

7. Acknowledgement

The author kindly thanks H. Alvarez, J. Frankland, D. Gruyer, D. Kresan, P. Morfouace, and O. Stezowski for providing insights, material and fruitful discussions during the preparation of this work.

References

[1] Blumenfeld Y, *Nucl. Instr. and Meth. B* **266 (2008) 4074–4079**
[2] Kimura K, *et al*, *Nucl.Instrum.Methods Phys.Res. A* **538 (2005) 608-614**
[3] York R C, *et al Proceedings of PAC09, Vancouver, BC, Canada,* **(2009) MG03GRI03**
[4] Dilling J, *et al*, *ISAC and ARIEL: The TRIUMF Radioactive Beam Facilities and the Scientific Program*, **ISBN 978-94-007-7963-1**
[5] Agostinelli S, *et al*, *Nucl. Instrum. Methods Phys. Res. A* **506 250–303 (2003)**
[6] Brun R and Rademakers F, *Nucl. Instrum. Methods Phys. Res. A* **389 81–6 (1997)**

[7] Frankland J, *et al*, "Kaliveda" [software], Available from https://github.com/kaliveda-dev/kaliveda [accessed 2022-09-16]
[8] Al-Turany M, *et al*, *J. Phys.: Conf. Ser.* **396 022001 (2012)**
[9] Al-Turany M, *et al*, *J. Phys.: Conf. Ser.* **513 022001 (2014)**
[10] Al-Turany M, *et al*, *FairRootGroup/FairRoot: v18.6.0(v18.6.0). Zenodo.* **https://doi.org/10.5281/zenodo.4588409** (2021).
[11] Matta A, *et al*, *J. Phys. G: Nucl. Part. Phys.* **43 045113 (2016)**
[12] Matta A *et al*, "nptool" [software], Version 3, Available from https://gitlab.in2p3.fr/np/nptool [accessed 2022-09-16]
[13] Parlog M, *et al*, *Nucl. Instr. and Meth. A* **613 (2010), 290-294**
[14] Böhlen T T, *et al*, *Nuclear Data Sheets* **120, 211-214 (2014)**
[15] Ferrari A, *et al*, *CERN-2005-10 (2005), INFN/TC_05/11, SLAC-R-773*
[16] Werner C J, *et al.*, *"MCNP6.2 Release Notes", Los Alamos National Laboratory,* **report LA-UR-18-20808 (2018)**
[17] Heuser J.M. for the CBM collaboration,*Nuclear Physics A* **830 (2009) 563c–566c**
[18] The PANDA Collaboration, *Eur. Phys. J. A* **52: 325 (2016)**
[19] CBM collaboration, "CMBRoot" [software], Available from https://git.cbm.gsi.de/computing/cbmroot [accessed 2022-09-16]
[20] Ship collaboration, "FairShip" [software], Available from https://github.com/ShipSoft/FairShip [accessed 2022-09-16]
[21] MDP collaboration, "MDPRoot" [software], Available from https://git.jinr.ru/nica/mpdroot/tree/dev [accessed 2022-09-16]
[22] Cabanelas P, *et al*, *J. Phys.: Conf. Ser.* **1024 012038 (2018)**
[23] Richter M, *AliceO2Group/AliceO2: First stable release (O2-1.0.0). Zenodo.* **https://doi.org/10.5281/zenodo.1493334 (2018)**
[24] R3B collaboration, "R3BRoot" [software], Available from github.com/R3BRootGroup/R3BRoot [accessed 2022-09-16]
[25] Stefano Spataro for the PANDA Collaboration, *J. Phys.: Conf. Ser.* **331 032031 (2011)**
[26] ATTPC collaboration, "ATTPCRoot" [software], Available from github.com/ATTPC [accessed 2022-09-16]
[27] Expert collaboration, "ExpertRoot" [software], Available from https://github.com/FLNR-JINR/er/ [accessed 2022-09-16]
[28] BMN collaboration, "BMNRoot" [software], Available from https://git.jinr.ru/nica/bmnroot [accessed 2022-09-16]
[29] Pollacco E, *et al*, *The European Physical Journal A - Hadrons and Nuclei* **25.1 (2005), pp. 287–288**
[30] Diget C Aa, *et al*, *Journal of Instrumentation*, **6 P02005 (2011)**
[31] Catford W N, *et al*, *AIP Conference Proceedings* **680, (2003), pp. 329-332.**
[32] Obertelli A, *et al*, *Eur. Phys. J. A* **(2014) 50: 8 DOI 10.1140/epja/i2014-14008-y**
[33] Mauss B, *et al*, *Nucl. Instr. and Meth. A* **940, (2019), 498-504**
[34] Matta A *et al*, "nptool" [software], Version 4, Available from https://gitlab.in2p3.fr/nptool/nptool [accessed 2022-09-16]