# Status of the FairRoot framework

*M. Al-Turany*[1], *D. Bertini*[1], *R. Karabowicz*[1], *D. Klein*[1], *D. Kresan*[1], *A. Lebedev*[1], *A. Manafov*[1], *A. Rybalchenko*[1], *T. Stockmanns*[2], *F. Uhlig*[1], *and N. Winckler*[1]

[1]GSI, Darmstadt, Germany; [2]FZJ, Juelich, Germany

## Introduction

Using FairRoot one can create simulated data and/or perform analysis within the same framework. The framework delivers base classes which enable the users to construct their detectors and/or analysis tasks in a simple way [1, 2]. The user analysis is organized in tasks (FairTask). Tasks are executed sequentially in the order they are added to the run manager. One way to improve the performance of FairRoot on multi-core processors is to run the independent tasks in parallel. Which can be done by running each task in a separate thread or as separate process. However, by multi-threading an error in one thread can bring down all the threads in the process where in multi-process the different processes are insulated from each other by the OS or even the network, i.e: An error in one process cannot bring down directly another one. On the other hand, the inter-thread communication is faster than interprocess one. Trying to find the correct balance between reliability and performance we decided to use the multi-process concept with message queues for data exchange, i.e: Each "Task" is a separate process, which can be also multithreaded, and the data exchange between the different tasks is done via messages. This concept allow us to create different topologies of tasks that can be adapted to the problem itself, and the hardware capabilities. Some of the advantages of such a system are:

- Flexibility: Different data paths can be modeled by simply changing the topology of Tasks.
- Scalability: Spread the work over several processes and machines on the fly.
- Adaptive: Sub-systems are continuously under development and improvement and can be exchanged individually.

## FairMQ: Base for data transport layer in FairRoot

Extending FairRoot to support multi-processes needs a solid and stable communication layer that should handle the whole communication in a transparent and stable way. Fortunately, such a layer exists already and is called ZeroMQ [3]. ZeroMQ is a very lightweight messaging system, specially designed for high throughput and low latency scenarios. It is open source, embeddable socket library that redefines the term socket as a general transport endpoint for atomic messages. ZeroMQ sockets provide efficient transport options for inter-thread, inter-process and inter-node communication (see Figure 1). Moreover it provides a Pragmatic General Multicast pattern (PGM), which is a reliable multicast protocol [3].

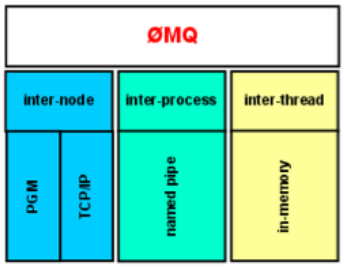


Figure 1: Message transport options in ZeroMQ. **Named Pipe**: Piece of random access memory (RAM) managed by the operating system and exposed to programs through a file descriptor and a named mount point in the file system. It behaves as a first in first out (FIFO) buffer

The zero in ZeroMQ refers to a culture of minimalism that permeates the project. They want to add power by removing complexity rather than exposing new functionality [4]. The library provides a built-in routing strategies for one-to-many or many-to-one communication scenarios. Each socket potentially comes with a sending and a receiving queue of configurable sizes.

The FairMQ package is described in more details by A. Rybalchenk in this report [5]

## Outlook and future plans

We need to design and develop a dynamic deployment system (DDS) that can utilize any resource management system (e.g: Slurm, GridEngine, ...etc.). A monitoring and logging system is also under development (See report by Anar Manafov [6]).

## References

[1] M. Al-Turany et al. The FairRoot framework *J. Phys.: Conf. Ser. 396 022001, 2012.*

[2] FairRoot: http://fairroot.gsi.de.

[3] http://www.zeromq.org/

[4] P. Hintjens, http://zguide.zeromq.org/page:all

[5] A. Rybalchenk and M. Al-Turany, Streaming data processing with FairMQ

[6] Anar Manfov, DDS: A New Dynamic Deployment System, this report