21st International Conference on Computing in High Energy and Nuclear Physics **CHEP2015** Okinawa Japan: April 13 - 17, 2015

Contribution ID: **367** Type: **poster presentation**

fwk: a go-based concurrent control framework

fwk: a go-based concurrent control framework

Current HEP control frameworks have been designed and written in the early 2000's, when multi-core architectures were not yet pervasive.
As a consequence, an inherently sequential event processing design emerged.
Evolving current frameworks' APIs and data models encouraging global states, non-reentrancy and non-thread-safety to a more concurrent friendly environment, in an adiabatic way, is a major undertaking, even more so when relying on the building blocks provided by C++.

This paper reports on the development of fwk, a framework investigating and leveraging the built-in tools of the Go language to enable concurrency at the event- and sub-event-levels. The concurrency features, code distribution and tooling of Go will be first presented to set the scene and explain why Go -and its ecosystem at large- is a natural fit for a concurrent framework.
The aim for such a framework is to be usable as a big HEP experiment's control framework. fwk is also meant to be a nimble application for analyses which require quick development/deployment cycles but without paying the price of a VM on the runtime performances side.

The paper will then discuss the design decisions applied to the various components of fwk, introduce its current capabilities (I/O, histogramming, dataflow, ...) and highlight how the design decisions of the Go language adequately paved the way for fwk.
Then, fads, a fast detector simulation toolkit, will be introduced. 'fads' is the port of a single-threaded fast detector simulation toolkit (Delphes) to fwk.
The paper will present benchmarks (CPU, RSS, I/O, scalability) of real Delphes use cases against fads, as well as comparisons of fads performances with regard to other next-generation concurrent frameworks, using synthetic data.
Performance tools and concurrency debugging tools provided with the Go toolchain and used for this comparison will also be presented.

Finally, the paper will present prospects and prototypes to fill the gap in the nascent Go-based HEP ecosystem and landscape - namely: histograms interactive displays and interactive analysis tools.

Primary author: Dr BINET, Sebastien (IN2P3/LAL)

Presenter: Dr BINET, Sebastien (IN2P3/LAL)

Track Classification: Track2: Offline software