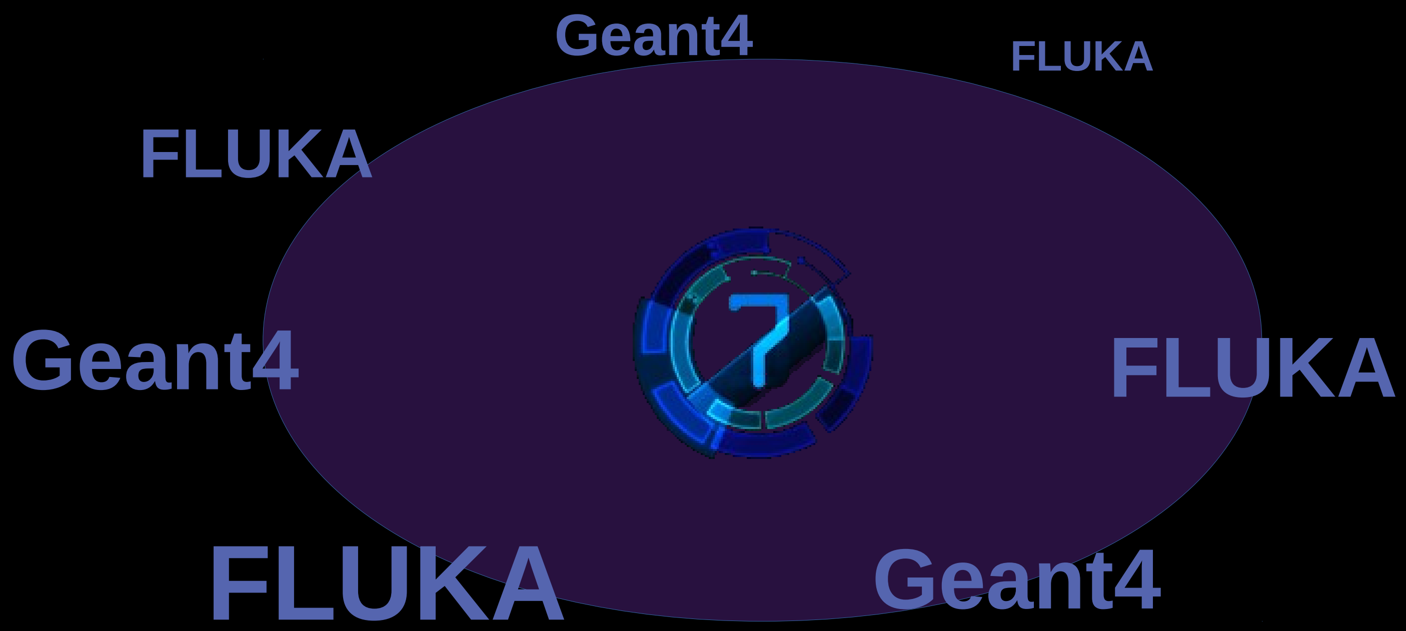


Geant4

FLUKA

FLUKA

Geant4

FLUKA

FLUKA

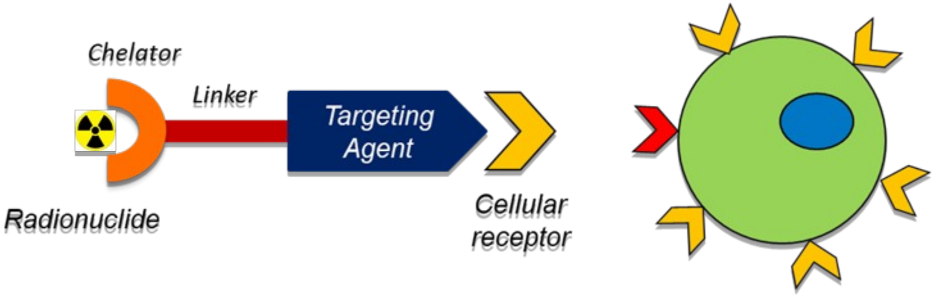
Geant4

The ISOLPHARM approach for parallelizing FLUKA and Geant4

Lisa Zangrando

The ISOLPHARM project

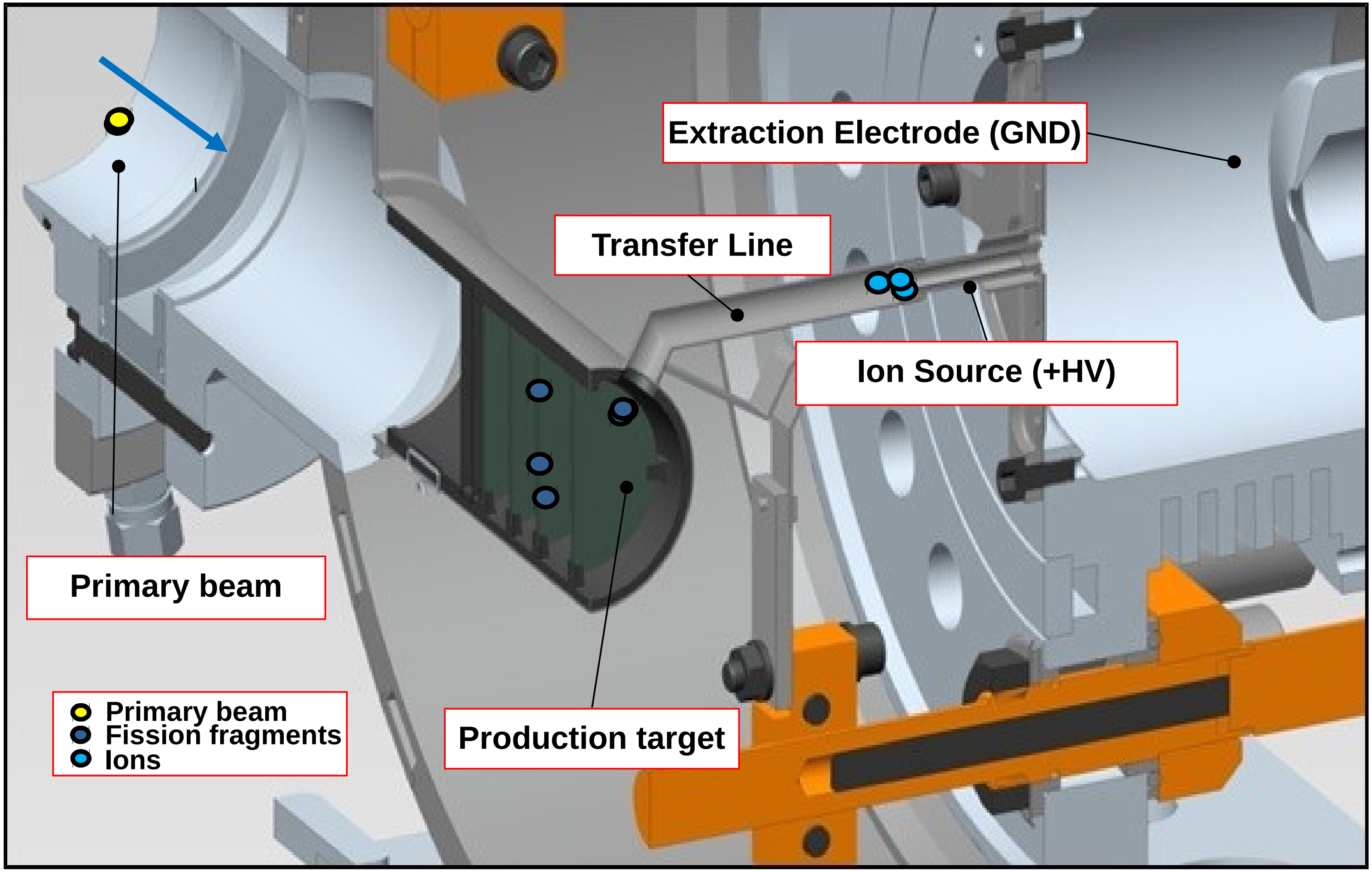
- ISOLPHARM is an INFN project devoted to the discovery and development of high purity radiopharmaceuticals which are drugs delivering a *radionuclide* to a specific cancer cell allowing diagnostic or therapeutic procedures



- In particular ISOLPHARM_Ag is studying the promising silver isotope (111Ag)

- Radionuclides will be produced by using the SPES facility (synchrotron) at INFN-LNL

The production target



The computing needs

- It is extremely important to study the behavior of the production target

- We need to predict the production and release of 111Ag from the primary target

- We achieve it by executing Monte Carlo simulations based on
  - **FLUKA: to simulate the production inside the primary target**
  - **Geant4: to simulate the diffusion and effusion processes**

- An IT infrastructure has been designed to meet the project computing needs:
  - **Common distributed environment for executing FLUKA and Geant4 simulations**
    - both of them are computing intensive and must be parallelized
  - Web based portal to allow users to:
    - create, submit and control their simulations
    - collect, view and analyze the produced data

Geant4 vs FLUKA

	Geant4	FLUKA
Multi-core CPU	multithreading	no multithreading one FLUKA instance per core
Distributed computing (extra setup needed)	ParGeant4 (MPI)	HTC cluster (i.e. HTCondor)
deployment	**Docker**	**Docker**

Toward a common approach

- Since FLUKA and Geant4 are containerized (Docker) applications
  - small, fast and portable runtime environments
  - use the OS-level virtualization
  - isolated from each other and from the host
  - microservice architecture

- their natural execution environment is a distributed platform (cluster) managed by a container **orchestrator** as Apache Mesos, Docker swarm, **Kubernetes**
  - open-source systems that manage large computing cluster
  - main features: automating deployment, auto-scaling, high availability, monitoring and management of containerized applications (i.e. orchestration).

- Kubernetes has been designed by Google and is the standard de facto

The ISOLPHARM approach

- A Monte Carlo simulation is composed of a set of FLUKA or Geant4 docker instances running on a Kubernetes cluster

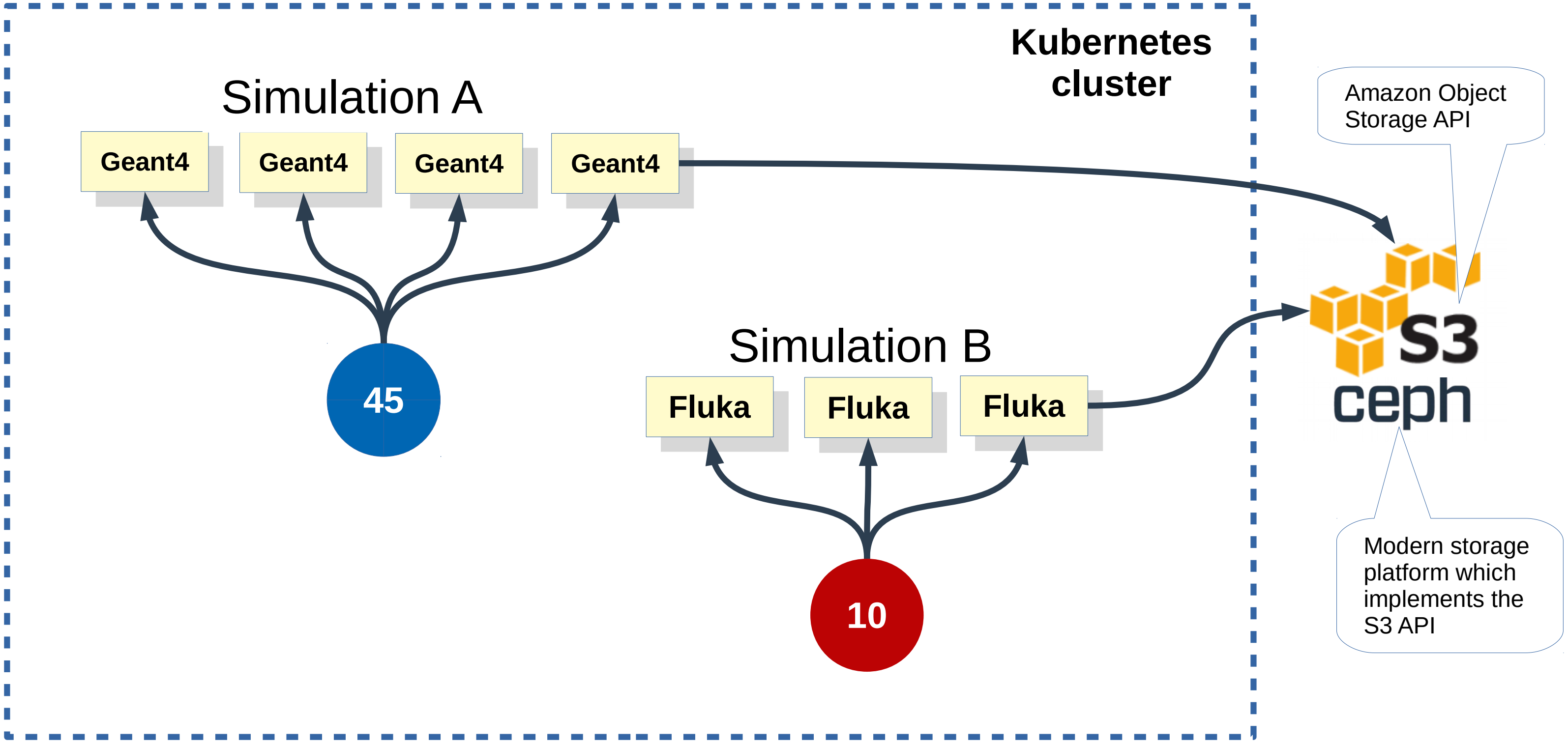
- More than one simulation can be executed at the same time

- all docker instances of the same simulation share:
  - the same input data
  - a countdown, an integer which indicates the overall number of runs/cycles to be done

- whenever an instance executes a new run it:
  - decrements the countdown by 1
  - **generates a new random seed number**

- as soon as the counter becomes zero, all related instances are automatically deleted and the partial files are merged and transferred to the storage

The computing architecture



Simulation input parameters

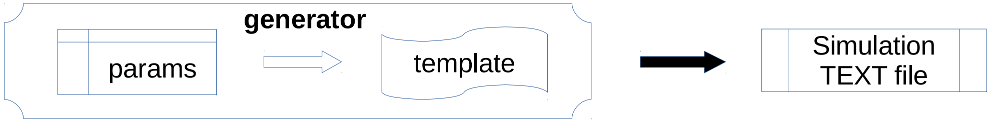
- Setting up a simulation is a very complex and error prone task

- The complexity increases if you consider that FLUKA and Geant4 have different input/output file formats
  - a standard would be appreciated

- To simplify the simulation set-up for normal (i.e. not advanced) users or scripts, we defined a set of abstract input parameters and specific simulation templates for FLUKA and Geant4

- A generator running inside the docker instance will apply such parameters to the proper template for generating the input file

Parameters					Production simulation		Release simulation
Topic	Parameter name		Variable Type	Unif of Measure	FLUKA	Standard G4	Custom G4
Target	Target material	Material type	string	-	Yes	Yes	Yes
		Target density	double	g/cm3	Yes	Yes	Yes
	Target temperature		array<double>[disk number]	°C	No	No	Yes
	Target geometry	Disk number	int	-	Yes	Yes	Yes
		Disk position	array<double>[disk number]	cm	Yes	Yes	Yes
		Disk radius	double	cm	Yes	Yes	Yes
		Disk thickness	array<double>[disk number]	cm	Yes	Yes	Yes
		Target box init	double	cm	Yes	Yes	Yes
		Target box end	double	cm	Yes	Yes	Yes
Beam	Particle (beam)		string	-	Yes	Yes	No
	Energy		double	MeV	Yes	Yes	No
	Beam center position (horizontal)		double	cm	Yes	Yes	No
	Beam center position (vertical)		double	cm	Yes	Yes	No
	beam position rms horizontal		double	cm	Yes	Yes	No
	beam position rms vertical		double	cm	Yes	Yes	No
	beam divergence rms horizontal		double	mrad	Yes	Yes	No
	beam divergence rms vertical		double	mrad	Yes	Yes	No
	beam current		double	mA	Yes	Yes	No
Yelds	Radioactive particle input fle		string	-	No	No	Yes
Other	Physics list		string	-	No	Yes	Yes
	Precision		double	-	Yes	Yes	Yes
	Random generator seeds		array<int>[2]	-	Yes	Yes	Yes



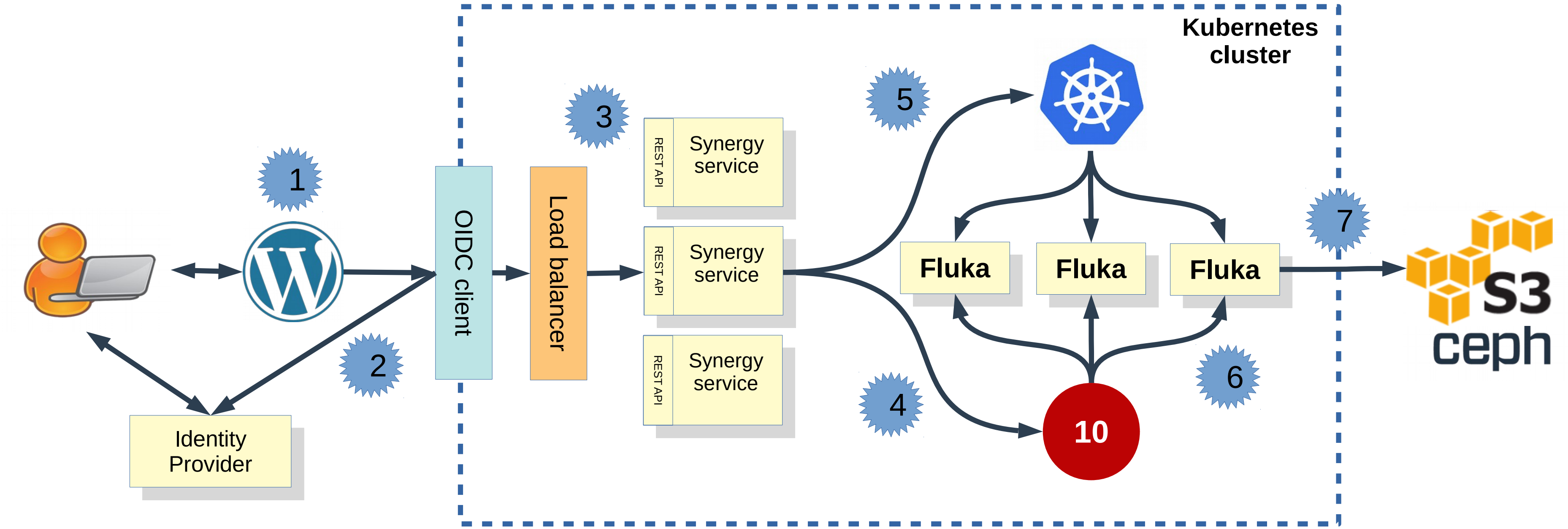
- Expert users can still use their input files accessible by a URL

The security

- Kubernetes supports several AuthN technologies (OIDC, SAML, Keystone, etc)
  - OIDC: Google, Facebook, Microsoft, etc
  - Keystone: OpenStack

- AuthZ based on Kubernetes RBAC (Role-Based Access Control)
  - local policies: tested

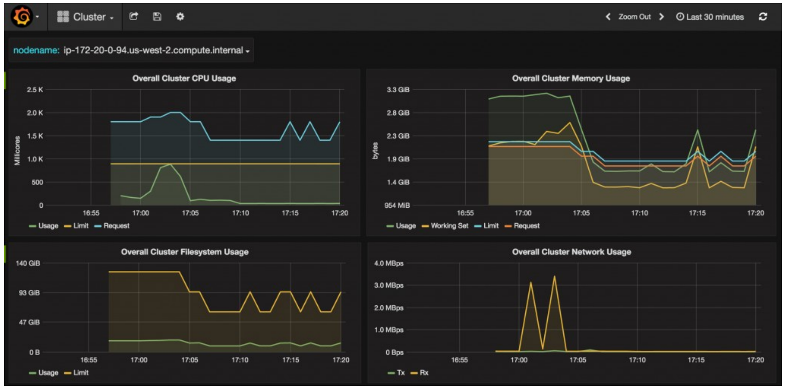
The full computing architecture



1) A user submits a simulation job through the Web Portal or CLI
2) Security: AuthN / AuthZ
3) LB forwards the job to Synergy (3 replicas in HA)
4) Synergy validates the user job request, creates the countdown in Redis
5) Synergy creates a new parallel job (FLUKA or Geant4 docker instances) in Kubernetes
6) Each instance generates its input file and decrement the coundown by 1 at each run/cycles
7) The output files produced by the task execution are stored into a S3 storage cluster

Monitoring with Prometheus

- Prometheus is an open source toolkit to monitor and alert, inspired by Google Borg Monitor
- Great solution to monitor a Kubernetes infrastructure and applications
- It has support for instrument applications (e.g. Python, Java, GO, etc)
  - custom metrics
- Instrumented our FLUKA docker image in order to publish the progress state
  - by parsing the FLUKA logs into the temp dir
- we are able to see the overall simulation state graphically shown as an histogram into the Grafana dashboard.

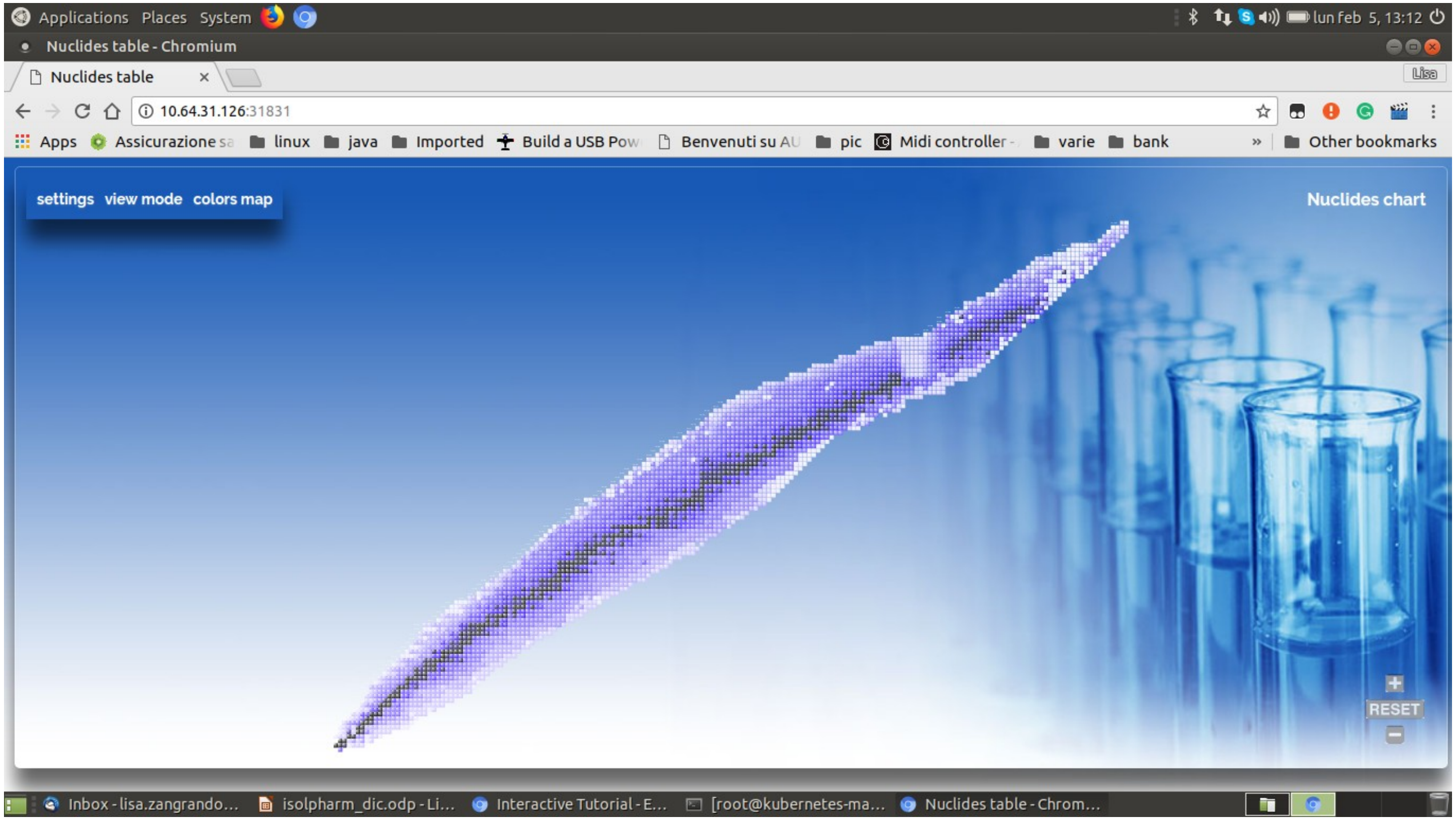


nodename: ip-172-20-0-94.us-west-2.compute.internal

The Web portal

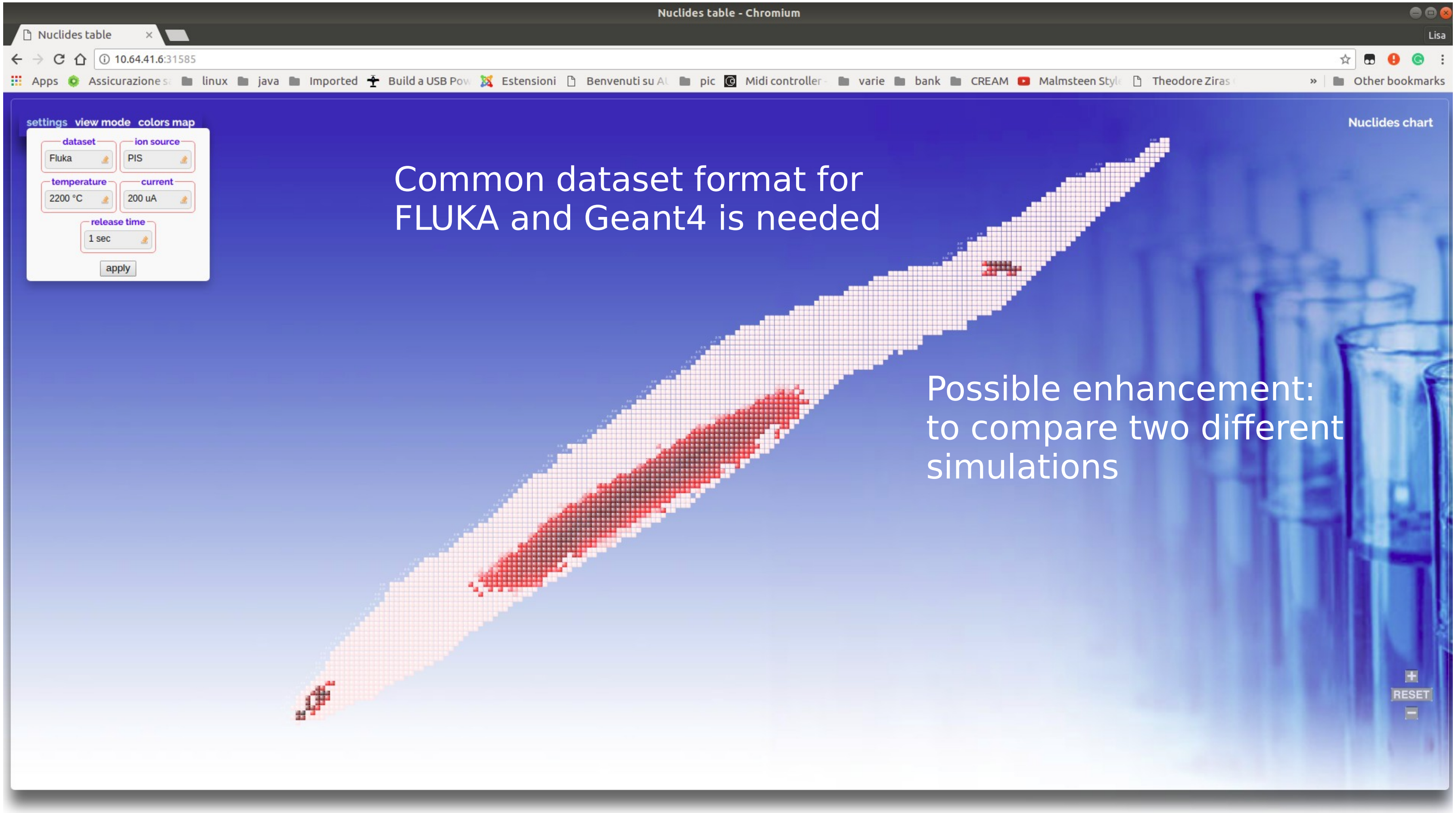
We are developing a Web portal based on WordPress, an open-source Content Management System (CMS) for publishing Web content

Implemented first prototype for graphically display the Geant4 and FLUKA simulation results (e.g. particles productions) mapped on the nuclides chart



The Web portal: the Nuclides Chart

Several view modes: by half life, particles, current (FC)



dataset: Fluka

ion source: PIS

temperature: 2200 °C

current: 200 uA

release time: 1 sec

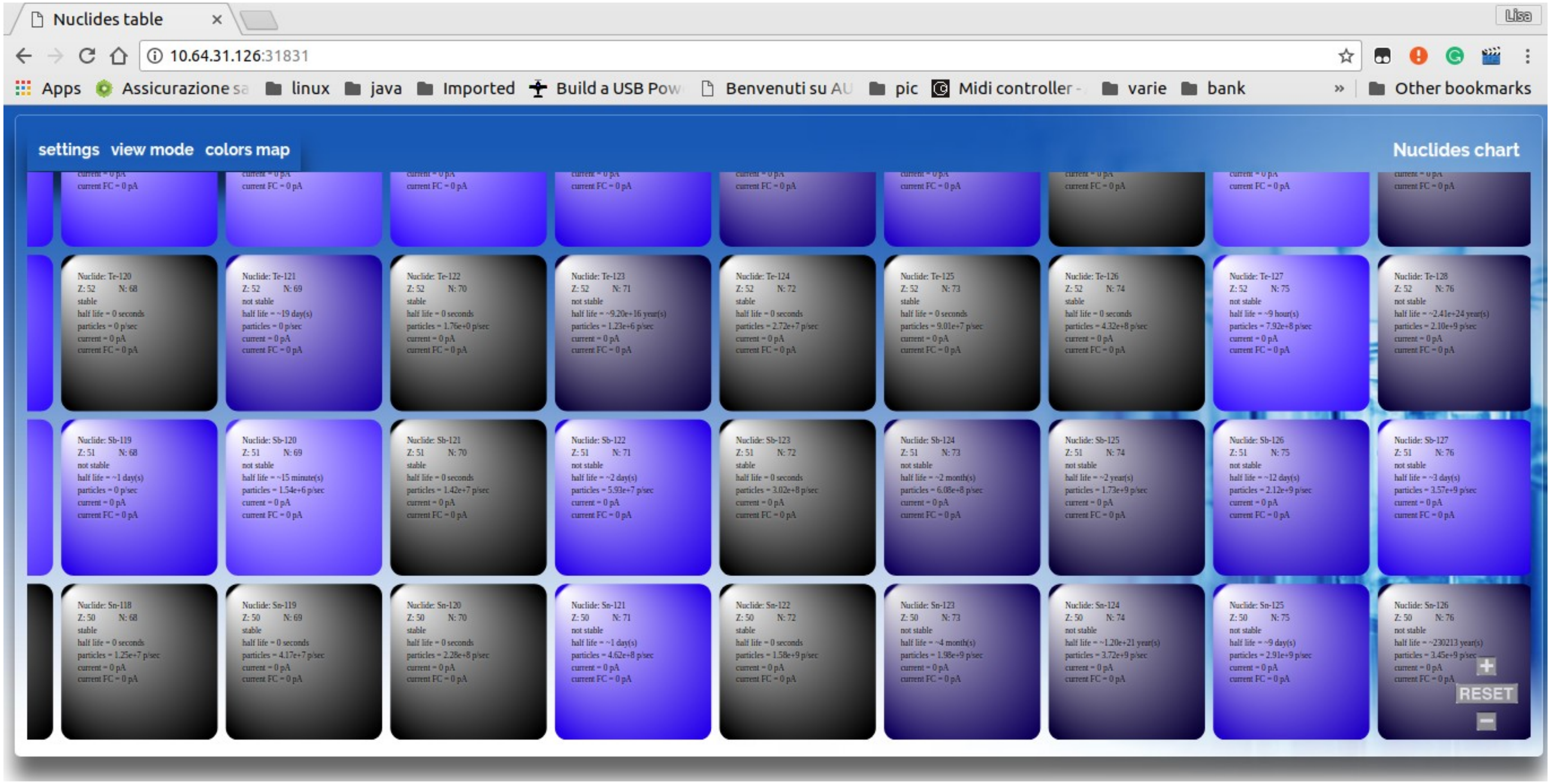
apply

Common dataset format for FLUKA and Geant4 is needed

Possible enhancement: to compare two different simulations

The Web portal: the Nuclides Chart

- The chart shows the detailed data about the produced nuclides
  - It is zoomable



current FC = 0 pA	current FC = 0 pA	current FC = 0 pA	current FC = 0 pA	current FC = 0 pA	current FC = 0 pA	current FC = 0 pA	current FC = 0 pA	current FC = 0 pA
Nuclide: Te-120 Z: 52 N: 68 stable half life = 0 seconds particles = 0 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-121 Z: 52 N: 69 not stable half life = ~19 day(s) particles = 0 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-122 Z: 52 N: 70 stable half life = 0 seconds particles = 1.76e+0 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-123 Z: 52 N: 71 not stable half life = ~9.20e+16 year(s) particles = 1.23e+6 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-124 Z: 52 N: 72 stable half life = 0 seconds particles = 2.72e+7 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-125 Z: 52 N: 73 stable half life = 0 seconds particles = 9.01e+7 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-126 Z: 52 N: 74 stable half life = 0 seconds particles = 4.32e+8 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-127 Z: 52 N: 75 not stable half life = ~9 hour(s) particles = 7.92e+8 p/sec current = 0 pA current FC = 0 pA	Nuclide: Te-128 Z: 52 N: 76 not stable half life = ~2.41e+24 year(s) particles = 2.10e+9 p/sec current = 0 pA current FC = 0 pA
Nuclide: Sb-119 Z: 51 N: 68 not stable half life = ~1 day(s) particles = 0 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-120 Z: 51 N: 69 not stable half life = ~15 minute(s) particles = 1.54e+6 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-121 Z: 51 N: 70 stable half life = 0 seconds particles = 1.42e+7 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-122 Z: 51 N: 71 not stable half life = ~2 day(s) particles = 5.93e+7 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-123 Z: 51 N: 72 stable half life = 0 seconds particles = 3.02e+8 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-124 Z: 51 N: 73 not stable half life = ~2 month(s) particles = 6.08e+8 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-125 Z: 51 N: 74 not stable half life = ~2 year(s) particles = 1.73e+9 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-126 Z: 51 N: 75 not stable half life = ~12 day(s) particles = 2.12e+9 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sb-127 Z: 51 N: 76 not stable half life = ~3 day(s) particles = 3.57e+9 p/sec current = 0 pA current FC = 0 pA
Nuclide: Sn-118 Z: 50 N: 68 stable half life = 0 seconds particles = 1.25e+7 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-119 Z: 50 N: 69 stable half life = 0 seconds particles = 4.17e+7 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-120 Z: 50 N: 70 stable half life = 0 seconds particles = 2.28e+8 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-121 Z: 50 N: 71 not stable half life = ~1 day(s) particles = 4.62e+8 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-122 Z: 50 N: 72 stable half life = 0 seconds particles = 1.58e+9 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-123 Z: 50 N: 73 not stable half life = ~4 month(s) particles = 1.98e+9 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-124 Z: 50 N: 74 not stable half life = ~1.20e+21 year(s) particles = 3.72e+9 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-125 Z: 50 N: 75 not stable half life = ~9 day(s) particles = 2.91e+9 p/sec current = 0 pA current FC = 0 pA	Nuclide: Sn-126 Z: 50 N: 76 not stable half life = ~230213 year(s) particles = 3.45e+9 p/sec current = 0 pA current FC = 0 pA

Kubernetes deployment

- we provide an automatic deployment system based on Ansible which is able create clusters on demand on:
  - OpenStack based Clouds
  - bare metal resources or VMs created on Cloud Providers other than OpenStack

- **basic topology: one master and at least two nodes**
  - https://github.com/zangrand/ansible-k8s

- the full Kubernetes cluster will be automatically provided in minutes
  - monitoring (Prometheus), dashboards (legacy, Grafana, etc), Big Data (Spark Operator)

- Kubernetes deployment in HA: on going

- **horizontal cluster autoscaling** is under investigation
  - the ability of the cluster to enlarge/shrink by itself as needed
  - speed up the simulation execution

The status

- Overall architecture validated
  - building block for Big Data analytics
- **Synergy (fronted) development: on going**
  - a different approach to be evaluated: implementing as Kubernetes operator (native API)
- FLUKA and Geant4:
  - docker images: available
  - common set of input parameters: available
  - simulation templates: FLUKA done, Geant4 on-going
  - monitoring for FLUKA: available
  - **monitoring for Geant4: under investigation**
- Storage: S3 support in Ceph: to be finalized
- Kubernetes deployment with Ansible: available
- **Web Portal based on WordPress: just evaluated**
  - **it would be nice to port Flair in WordPress**
- data analysis tools
  - first prototype of Nuclides chart to be ported in WordPress (common dataset to be defined)
  - some others to be implemented

Questions?