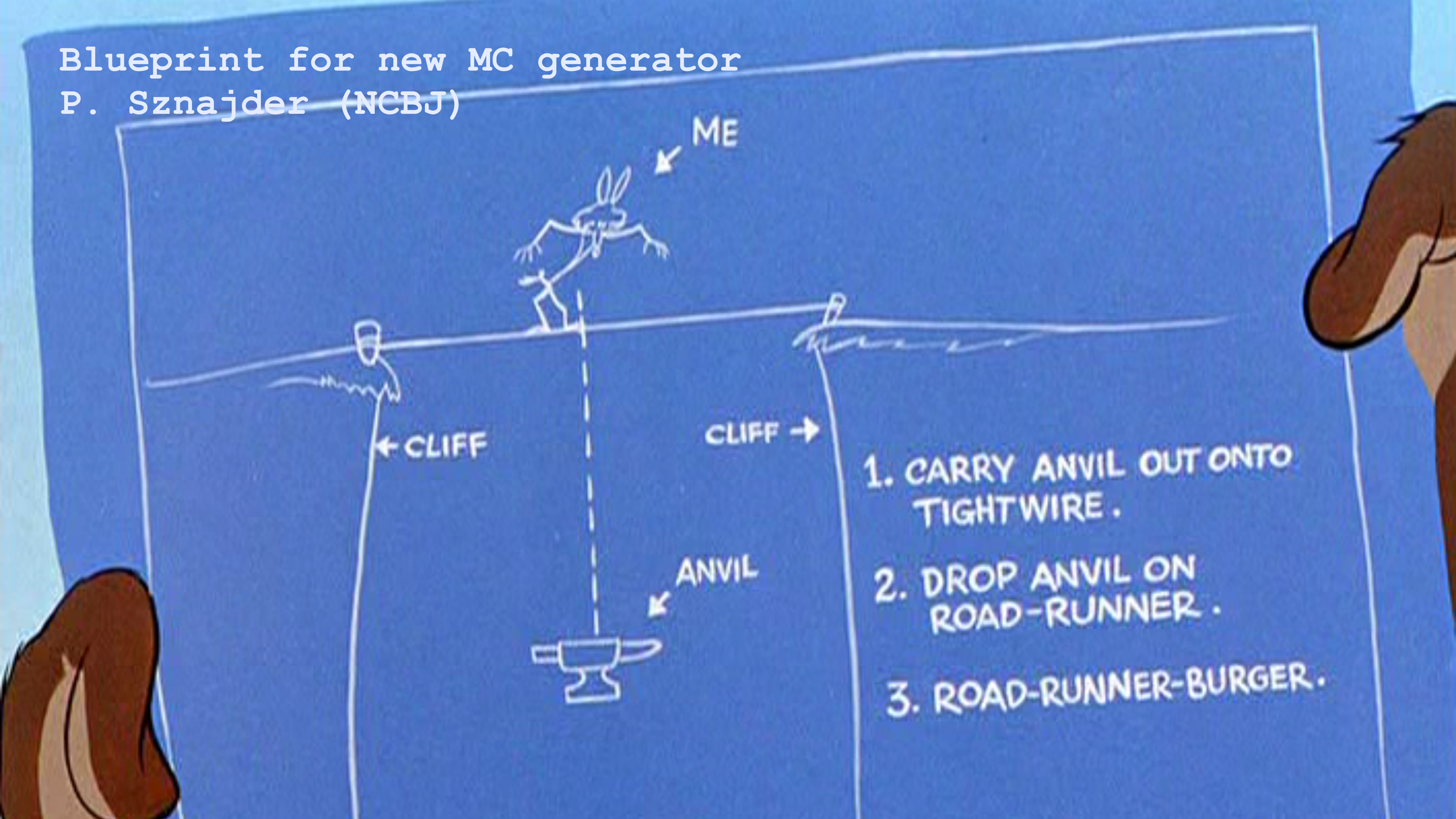
Blueprint for new MC generator

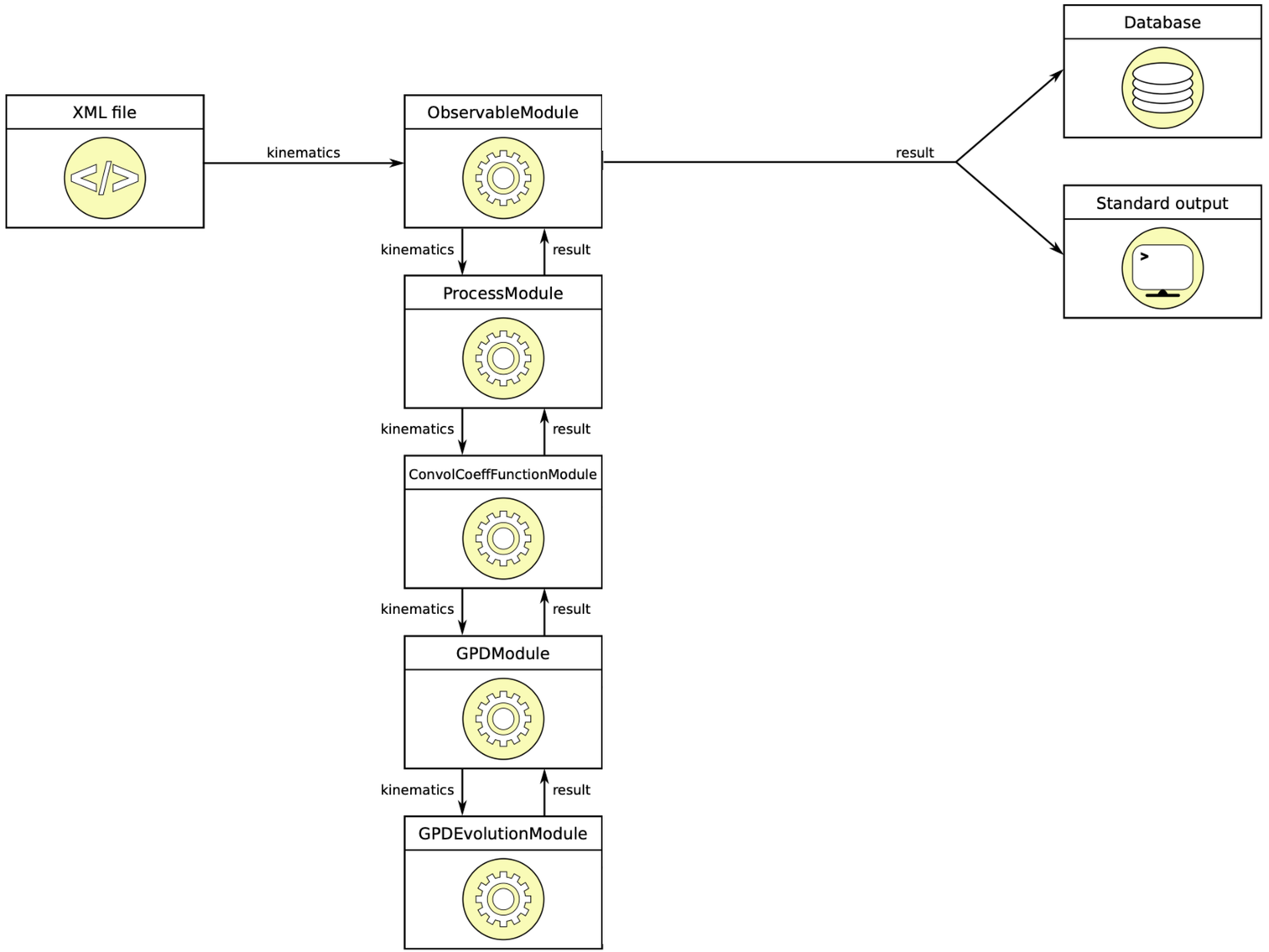
P. Sznajder (NCBJ)



Features

- Generic → to be used at CERN, JLab, EIC, … (no more “home-made” generators)
- Multichannel (DVCS, TCS, DVMP, …) capability → what is available in PARTONS will be available in new MC generator
- Modularity → for flexibility and simplicity
- Well developed → good coding practices, use modern programming language (c++) and paradigms
- Easy to maintain → we expect at least 20 years of lifetime, project must be well designed and properly established, must have useful documentation
- Robust
- Multithreaded (?)

PARTONS



Features

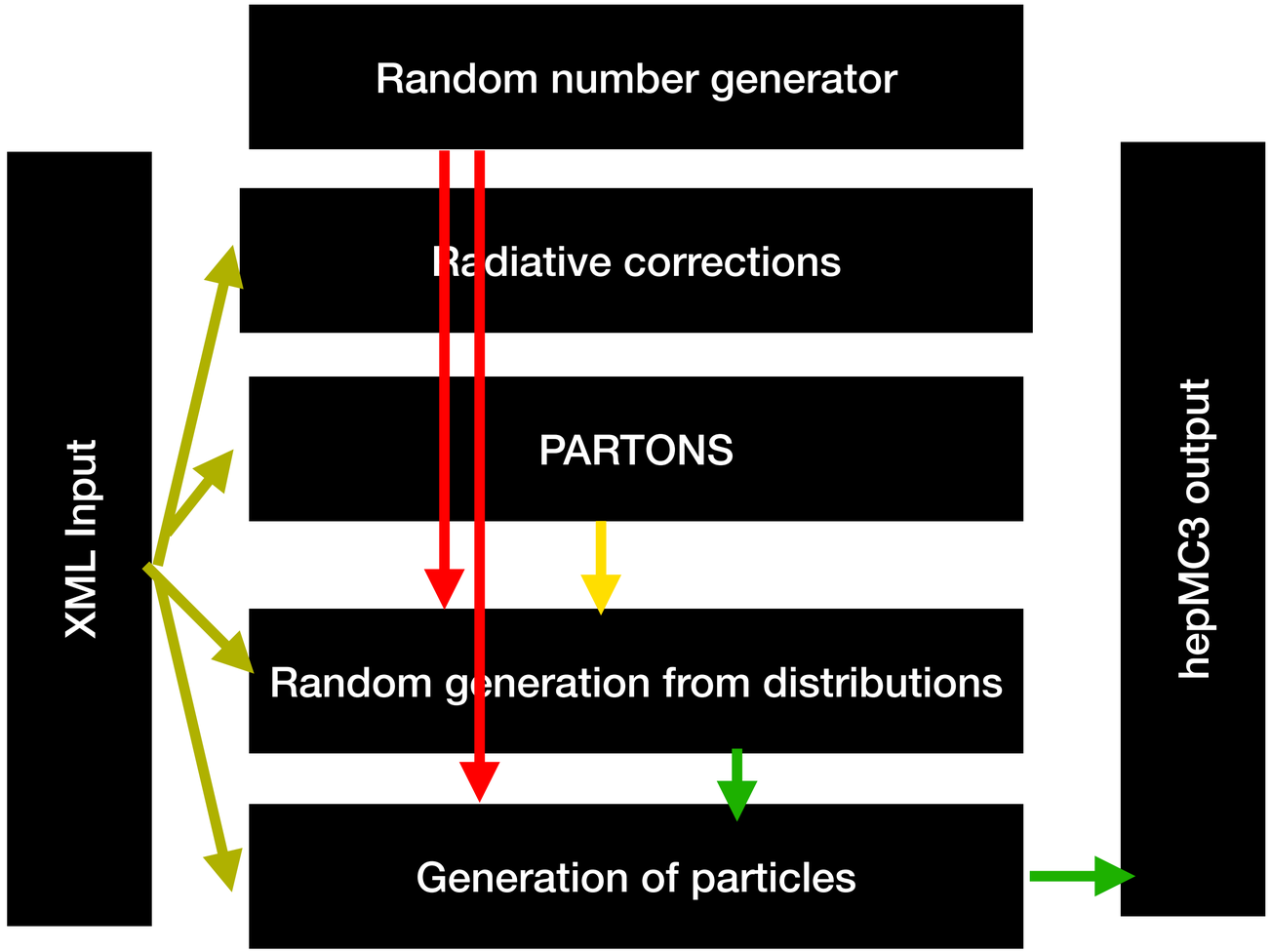
- modular
- well designed and maintained
- robust
- multichannel
- largest collection of GPD code available in one place
- multithreading
- use DB

PARTONS (example of scenario)

XML file

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>

<!--
This scenario demonstrates …
-->

<!-- Scenario starts here -->
<!-- For your convenience and for bookkeeping provide creation date and unique
description -->
<scenario date="2019-01-23" description="DVCS observable evaluation for single
kinematics example">

   <!-- First task: evaluate DVCS observable for a single kinematics -->
   <!-- Indicate service and its methods to be used and indicate if the result
   should be stored in the database -->
   <task service="ObservableService" method="computeObservable" storeInDB="0">

      <!-- Define DVCS observable kinematics -->
      <kinematics type="ObservableKinematic">
         <param name="xB" value="0.2" />
         <param name="t" value="-0.1" />
         <param name="Q2" value="2." />
         <param name="E" value="6." />
         <param name="phi" value="20." />
     </kinematics>

      <!-- Define physics assumptions -->
      <computation_configuration>

         <!-- Select DVCS observable -->
         <module type="Observable" name="DVCSAllMinus">

            <!-- Select DVCS process model -->
            <module type="ProcessModule" name="DVCSProcessBMJ12">

               <!-- Select scales module -->
               <!-- (it is used to evaluate factorization and renormalization
               scales out of kinematics) -->
               <module type="ScalesModule" name="ScalesQ2Multiplier">

                  <!-- Configure this module -->
                     <param name="lambda" value="1." />
                  </module>

                  <!-- Select xi-converter module -->
                  <!-- (it is used to evaluate GPD variable xi out of kinematics) -->
                  <module type="XiConverterModule" name="XiConverterXBToXi">
                  </module>

                  <!-- Select DVCS CFF model -->
                  <module type="ConvolCoeffFunctionModule" name="DVCSCFFStandard">

                     <!-- Indicate pQCD order of calculation -->
                     <param name="qcd_order_type" value="NLO" />

                     <!-- Select GPD model -->
                     <module type="GPDModule" name="GPDGK16Numerical">
                     </module>

                  </module>

               </module>

            </module>

         </computation_configuration>

      </task>

      <!-- Second task: print results of the last computation into standard
      output -->
      <task service="ObservableService" method="printResults">
      </task>

   </scenario>
```

New MC generator



What's to be done?

- cross-sections will be evaluated by PARTONS starting from GPD, amplitude or observable level
- we will need interpolation/tessellation methods to build models based on lookup tables (modularisation?), it should be possible to use DB to store lookup tables
- radiative corrections
- service for random number generator
- random generation from know distributions
- evaluation of 4momenta and decays
- interface via xml
- multithreading
- output in hepMC3 format

Closing remarks

- This will be the first general generic MC generator in GPD field
- Possible collaboration with EIC software group
- Name?