

Hadronic Developer Parameters

KOI, Tatsumi

Dotti, Andrea

Wright, H. Dennis

SLAC National Accelerator Laboratory

G4HadronicDeveloperParameters

- A utility class for handling hadronic developer parameters
 - Should be simple but provide reasonable functionalities
 - Set, Get, SetDefault, GetDefault and Dump
 - Allow to set lower and upper limit for a parameter
- Hadronic developer parameter is a parameter that a model developer opens to specific (experienced) user for changing the value
 - Model developer gives name of the parameter to the user.
 - The user changes the value of the parameter and checks its impact to final result.
- Type of parameter can be double, integer or Boolean.

For a parameter,

Model developer

- Must set default value of the parameter
 - Enable to register the lower and upper limits to the parameter
- Use a value through **DeveloperGet** method in the initialization of model
 - To be enable to issue a warning message, when the value is modified

User (tester)

- Can set a new value for the parameter
 - **Should do this very early in his/her code.**
 - **Changes only allow once**
- Can check default value
- Can dump default and new value of a parameter with the name of parameter
 - Dump require a parameter name as an argument.

Target user of HadronicDeveloperParameters

- An experienced user who want to see an impact of model parameter in final result.
- He/her has no interested in actual implementation of the model.
- Users (or developers) who have a knowledge of model implementation are not primary target
 - They can easily wrote a codes for their purpose

Why

- only allow to change parameter values once.
 - Incorporation timing of a new value into model is not well defined in hadronic framework
 - Minimize the risk that change of the parameter is not reflected in the calculation.
- dump always requires name of parameter
 - We do not want to provide a full list of developer parameter for users.

What is “DeveloperGet”

- This is the method model developer uses in incorporation of parameter values during the initialization of model.
- If the value is different from default, then warning message is issued.
 - If user changed a value but not having this warning message, then the new value is not in use in the simulation by some reasons. Most likely it was too late to set the parameter.
- Difference between “DeveloperGet” and “Get”
 - The later will not issue the warning messages.
 - The use case of the later is that a user want to know current parameter value by some reason, for example print out.

G4HadronicDeveloperParameters

source/processes/hadronic/util

```
class G4HadronicDeveloperParameters
{
public:
    static G4HadronicDeveloperParameters& GetInstance();
private:
    G4HadronicDeveloperParameters();

public:
    G4bool Set( const std::string name , const G4double );
    G4bool GetDefault( const std::string name , G4double& value );
    G4bool Get( const std::string name , G4double& value );
    G4bool SetDefault( const std::string name , const G4double value , G4double lower_limit = -DBL_MAX , G4double upper_limit = DBL_MAX );
    G4bool DeveloperGet( const std::string name , G4double& value );
    void Dump( const std::string name );

    """
}
```

Developer side implementation

G4CascadeParameters

```
#include "G4HadronicDeveloperParameters.hh"
#define OLD_RADIUS_UNITS (3.3836/1.2)      // Used with NucModel params
namespace {
    G4HadronicDeveloperParameters& HDP = G4HadronicDeveloperParameters::GetInstance();
    class BERTParameters {
        public:
            BERTParameters(){
                // Define default values
                //HDP.SetDefault("NAME",VALUE,LOWER_LIMIT(default=-DBL_MAX),UPPER_LIMIT(default=DBL_MAX));
                HDP.SetDefault( "BERT_RADIUS_SCALE" , OLD_RADIUS_UNITS );
                HDP.SetDefault( "BERT_XSEC_SCALE" , 1.0 , 0. );
            }
        };
        BERTParameters BP;
    }

void G4CascadeParameters::Initialize() {
    """
    //RADIUS_SCALE = (G4NUCMODEL_RAD_SCALE ? strtod(G4NUCMODEL_RAD_SCALE,0)
    //           : (BEST_PAR?1.0:OLD_RADIUS_UNITS));
    HDP.DeveloperGet("BERT_RADIUS_SCALE",RADIUS_SCALE);

    """
    //XSEC_SCALE = (G4NUCMODEL_XSEC_SCALE ? strtod(G4NUCMODEL_XSEC_SCALE,0)
    //           : (BEST_PAR?0.1:1.0));
    HDP.DeveloperGet("BERT_XSEC_SCALE",XSEC_SCALE);
}
}
```

User (tester) side implementation

User main (Hadr01)

```
#include "G4HadronicDeveloperParameters.hh"
int main(,,,)
{
//followings should be implemented very early
G4HadronicDeveloperParameters& HDP =
G4HadronicDeveloperParameters::GetInstance();
    HDP.Set("BERT_RADIUS_SCALE",1.0);
//HDP.Set("BERT_RADIUS_SCALE",1.0); //cause error
    HDP.Set("BERT_XSEC_SCALE",2.0);
    HDP.Dump("BERT_RADIUS_SCALE");
    HDP.Dump("BERT_XSEC_SCALE");

"""
}
```

In run time, you should have two warning messages
for BERT_RADIUS_SCALE and BERT_XSEC_SCALE

Status

- G4HadronicDeveloperParameters
 - committing tags, but the last one is still rejected because there is warning message in Windows platform.
- G4CascadeParameters
 - will commit after acceptance of HadronicDeveloperParameters tag.
 - Applying activation key, for a while.
- Only usage of “double” type parameter is tested through BERT-like cascade model.

Restriction that parameters can be changed only once in a job

- Is this limitation too much strict?
 - User may need to change multiple times in a job
 - Is there any fundamental reason why it can not be divided into multiple jobs?
- Hadronic framework does not define solid incorporation timing of parameters into simulation (model).
 - We cannot guarantee that new values are properly incorporated in later calculation.
 - Some models can handle properly but others are not

Future works

- Fix warning messages in Windows platform and make available current version to developers for testing
 - Next slide shows the warning, if you know how to fix it, please let me know!
- Usage of Boolean and integer parameter will be checked through Precompound model

Warning messages from Windows platform

G4MuonMinusCapture.cc

G4HadronicAbsorptionBertini.cc

G4MuonMinusAtomicCapture.cc

G4Bessel.cc

G4HadFinalState.cc

G4HadProjectile.cc

G4HadSecondary.cc

G4HadSignalHandler.cc

G4HadTmpUtil.cc

G4HadronicDeveloperParameters.cc

C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\include\utility(158): warning C4800: 'const G4int': forcing value to bool 'true' or 'false' (performance warning) [D:\jk\workspace\g4-win_2\VC14\win7vs14\Platform\x86\Seq\nightly\RelWithDebInfo\g4tags-dev\build\source\processes_G4processes-archive.vcxproj]

C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\include\xmemory0(657): note: see reference to function template instantiation 'std::pair<const _Kty, _Ty>::pair<std::string, const G4int, void>(std::pair<std::string, const G4int> &&) noexcept' being compiled with [_Kty=std::string, _Ty=const G4bool]

C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\include\xmemory0(657): note: see reference to function template instantiation 'std::pair<const _Kty, _Ty>::pair<std::string, const G4int, void>(std::pair<std::string, const G4int> &&) noexcept' being compiled with [_Kty=std::string,