```
//
// ********************************************************************
// * License and Disclaimer                                           *
// *                                                                  *
// * The  Geant4 software  is  copyright of the Copyright Holders  of *
// * the Geant4 Collaboration.  It is provided  under  the terms  and *
// * conditions of the Geant4 Software License,  included in the file *
// * LICENSE and available at  http://cern.ch/geant4/license .  These *
// * include a list of copyright holders.                             *
// *                                                                  *
// * Neither the authors of this software system, nor their employing *
// * institutes,nor the agencies providing financial support for this *
// * work  make  any representation or  warranty, express or implied, *
// * regarding  this  software system or assume any liability for its *
// * use.  Please see the license in the file  LICENSE  and URL above *
// * for the full disclaimer and the limitation of liability.         *
// *                                                                  *
// * This  code  implementation is the result of  the  scientific and *
// * technical work of the GEANT4 collaboration.                      *
// * By using,  copying,  modifying or  distributing the software (or *
// * any work based  on the software)  you  agree  to acknowledge its *
// * use  in  resulting  scientific  publications,  and indicate your *
// * acceptance of all terms of the Geant4 Software license.          *
// ********************************************************************
//
//          Filippo Ambroglini : filippo.ambroglini@pg.infn.it
//
//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......
//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......

#include "HEPDSWDetectorConstruction.hh"
#include "HEPDSWDetectorMessenger.hh"

#include "HEPDSWMaterial.hh"
#include "G4Box.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVReplica.hh"
#include "G4VisAttributes.hh"

#include "TrackerConstruction.hh"
//#include "ScintillatorConstruction.hh"
#include "CalorimeterConstruction.hh"
#include "HEPDBoxConstruction.hh"
#include "SatelliteConstruction.hh"


#include "G4GeometryManager.hh"
#include "G4PhysicalVolumeStore.hh"
#include "G4LogicalVolumeStore.hh"
#include "G4SolidStore.hh"

#include "G4UImanager.hh"
#include "G4UnitsTable.hh"
#include "G4PhysicalConstants.hh"
#include "G4SystemOfUnits.hh"
#include <iomanip>

#include "G4SDManager.hh"
#include "MCTruthSD.hh"

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......

HEPDSWDetectorConstruction::HEPDSWDetectorConstruction()
  :fSolidWorld(0),fLogicWorld(0),fPhysiWorld(0),
   fSatelliteBuilder(0),fHEPDBoxBuilder(0),fCaloBuilder(0),fTrackerBuilder(0)

{
```



```
  pMaterial = new HEPDSWMaterial();
  fDetectorMessenger = new HEPDSWDetectorMessenger(this);

  fworldHalfX=20.0*cm;
  fworldHalfY=20.0*cm;
  fworldHalfZ=30.0*cm;

  fSatelliteBuilder = new SatelliteConstruction();
  fHEPDBoxBuilder   = new HEPDBoxConstruction();
  fCaloBuilder      = new CalorimeterConstruction();
  //  fScintBuilder     = new ScintillatorConstruction();
  fTrackerBuilder   = new TrackerConstruction();

  useSatellite=true;
  useHEPDBox=true;
  useCalorimeter=true;
  //  useScintillator=true;
  useTracker=true;
  theSatelliteConfig="Config2";
  theHEPDBoxConfig="Config2";
  theCaloConfig="Config6";
  //theScintConfig="Config4";
  theTrackerConfig="Config2";
}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......

HEPDSWDetectorConstruction::~HEPDSWDetectorConstruction()
{
  if (pMaterial) delete pMaterial;
  if (fDetectorMessenger) delete fDetectorMessenger;
}

void HEPDSWDetectorConstruction::SetWorldDimensions(G4double aHalfX,G4double aHa
lfY,G4double aHalfZ)
{
  if(fworldHalfX!=aHalfX||fworldHalfY!=aHalfY||fworldHalfZ!=aHalfZ){
    fworldHalfX=aHalfX;
    fworldHalfY=aHalfY;
    fworldHalfZ=aHalfZ;
  }
}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......

G4VPhysicalVolume* HEPDSWDetectorConstruction::Construct()
{

  G4SDManager* SDman = G4SDManager::GetSDMpointer();
  G4String mcTruthSDname = "/hepd/mctruth";
  MCTruthSD* mcSD = new MCTruthSD(mcTruthSDname);
  SDman->AddNewDetector(mcSD);

  pMaterial -> DefineMaterials();
  G4Material* vacuum = pMaterial->GetMaterial("Galactic");


  fSolidWorld = new G4Box("world",fworldHalfX,fworldHalfY,fworldHalfZ);
  fLogicWorld = new G4LogicalVolume(fSolidWorld,vacuum,"world");

  fLogicWorld->SetSensitiveDetector(mcSD);

  fPhysiWorld = new G4PVPlacement(0,G4ThreeVector(),"world",fLogicWorld,0,false,0
);

  G4VisAttributes * attInvisible = new G4VisAttributes();
  attInvisible->SetVisibility(false);
  attInvisible->SetForceAuxEdgeVisible(false);
  fLogicWorld->SetVisAttributes(attInvisible);
```

```
  if(useSatellite)
    fSatelliteBuilder->Builder(theSatelliteConfig,fPhysiWorld);
  if(useHEPDBox)
    fHEPDBoxBuilder->Builder(theHEPDBoxConfig,fPhysiWorld);
  if(useCalorimeter)
    fCaloBuilder->Builder(theCaloConfig,fPhysiWorld);
//   if(useScintillator)
//     fScintBuilder->Builder(theScintConfig,fPhysiWorld);
  if(useTracker)
    fTrackerBuilder->Builder(theTrackerConfig,fPhysiWorld);
  return fPhysiWorld;
}

// void HEPDSWDetectorConstruction::CaloSetCaloMaterial(G4String aMat){
//   fCaloBuilder->SetCaloMaterial(aMat);
// }

void HEPDSWDetectorConstruction::CaloSetCaloMaterial(G4String aMat1,G4String aMa
t2){
  fCaloBuilder->SetCaloMaterial(aMat1,aMat2);
}

void HEPDSWDetectorConstruction::CaloSetVetoMaterial(G4String aMat){
  fCaloBuilder->SetVetoMaterial(aMat);
}

void HEPDSWDetectorConstruction::CaloSetPoronMaterial(G4String aMat){
  fCaloBuilder->SetPoronMaterial(aMat);
}

void HEPDSWDetectorConstruction::CaloSetCarbonFiberMaterial(G4String aMat){
  fCaloBuilder->SetCarbonFiberMaterial(aMat);
}

void HEPDSWDetectorConstruction::CaloSetHoneyCombMaterial(G4String aMat){
  fCaloBuilder->SetHoneyCombMaterial(aMat);
}

void HEPDSWDetectorConstruction::CaloSetNumberOfCrystalLayer(G4int aVal){
  fCaloBuilder->SetNumberOfCrystalLayer(aVal);
}

// void HEPDSWDetectorConstruction::ScintillatorSetScintillatorMaterial(G4String
 aMat){
//   fScintBuilder->SetScintillatorMaterial(aMat);
// }

// void HEPDSWDetectorConstruction::ScintillatorSetPoronMaterial(G4String aMat){
//   fScintBuilder->SetPoronMaterial(aMat);
// }

// void HEPDSWDetectorConstruction::ScintillatorSetCarbonFiberMaterial(G4String
aMat){
//   fScintBuilder->SetCarbonFiberMaterial(aMat);
// }

void HEPDSWDetectorConstruction::TrackerSetSiliconMaterial(G4String aMat){
  fTrackerBuilder->SetSiliconMaterial(aMat);
}

void HEPDSWDetectorConstruction::TrackerSetKaptonMaterial(G4String aMat){
  fTrackerBuilder->SetKaptonMaterial(aMat);
}

void HEPDSWDetectorConstruction::TrackerSetCarbonFiberMaterial(G4String aMat){
  fTrackerBuilder->SetCarbonFiberMaterial(aMat);
}
```



```
void HEPDSWDetectorConstruction::HEPDBoxSetBlanketMaterial(G4String aMat){
  fHEPDBoxBuilder->SetBlanketMaterial(aMat);
}

void HEPDSWDetectorConstruction::HEPDBoxSetBlanketMaterial(G4String aMat1,G4Stri
ng aMat2,G4String aMat3,G4String aMat4){
  fHEPDBoxBuilder->SetBlanketMaterial(aMat1,aMat2,aMat3,aMat4);
}

void HEPDSWDetectorConstruction::HEPDBoxSetWallMaterial(G4String aMat){
  fHEPDBoxBuilder->SetWallMaterial(aMat);
}

void HEPDSWDetectorConstruction::HEPDBoxSetWallMaterial(G4String aMat1, G4String
 aMat2){
  fHEPDBoxBuilder->SetWallMaterial(aMat1,aMat2);
}

void HEPDSWDetectorConstruction::SatelliteSetBlanketMaterial(G4String aMat){
  fSatelliteBuilder->SetBlanketMaterial(aMat);
}

void HEPDSWDetectorConstruction::SatelliteSetWallMaterial(G4String aMat){
  fSatelliteBuilder->SetWallMaterial(aMat);
}
```