```
//
// ********************************************************************
// * 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 "SatelliteConstructionConfig1.hh"


#include "G4NistManager.hh"
#include "HEPDSWMaterial.hh"
#include "G4Box.hh"
#include "G4SubtractionSolid.hh"
#include "G4LogicalVolume.hh"
#include "G4PVPlacement.hh"
#include "G4PVReplica.hh"

#include "G4GeometryManager.hh"
#include "G4PhysicalVolumeStore.hh"
#include "G4LogicalVolumeStore.hh"
#include "G4SolidStore.hh"
#include "G4VisAttributes.hh"
#include "G4SDManager.hh"

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

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

SatelliteConstructionConfig1::SatelliteConstructionConfig1()
  :fSolidBlanket(0),fSolidWall(0),
   fLogicBlanket(0),fLogicWall(0),
   fPhysiBlanket(0),fPhysiWall(0)
{
  pMaterial     = new HEPDSWMaterial();
  fBlanket_X  = 254*mm;
  fBlanket_Y  = 238*mm;
  fBlanket_Z  = 0.1*mm;

  fWall_X = 345*mm;
  fWall_Y = 490*mm;
  fWall_Z = 3.5*mm;
```



```
  fWallHole_X = 237*mm;
  fWallHole_Y = 218*mm;
  fWallHole_Z = 4*mm;


  ComputeObjectsPositioning();

  // materials
  blanketMaterial        = "mylar";
  wallMaterial           = "Aluminium";
}

//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......
SatelliteConstructionConfig1::~SatelliteConstructionConfig1()
{
  if (pMaterial) delete pMaterial;
}

void SatelliteConstructionConfig1::ComputeObjectsPositioning(){

  transWallHole_X = fWall_X/2.-fWallHole_X/2.-68.5*mm;
  transWallHole_Y = fWall_Y/2.-fWallHole_Y/2.-44*mm;

  fPhysiBlanket_Z = 386.57*mm;

  fPhysiWall_X = +14.5*mm;
  fPhysiWall_Y = -92*mm;
  fPhysiWall_Z = fPhysiBlanket_Z+fBlanket_Z/2.+fWall_Z/2.;

}


void SatelliteConstructionConfig1::SetBlanketMaterial(G4String aMat){
  blanketMaterial=aMat;
}
void SatelliteConstructionConfig1::SetWallMaterial(G4String aMat){
  wallMaterial=aMat;
}
//....oooOOOOOooo........oooOOOOOooo........oooOOOOOooo........oooOOOOOooo......

void SatelliteConstructionConfig1::Builder(G4VPhysicalVolume* motherVolume)
{

  pMaterial->DefineMaterials();
  G4Material* blanketMat = pMaterial->GetMaterial(blanketMaterial);
  G4Material* wallMat = pMaterial->GetMaterial(wallMaterial);

  G4RotationMatrix* myRot = new G4RotationMatrix;

  fSolidBlanket = new G4Box("fSolidThermalBlanket",fBlanket_X/2.,fBlanket_Y/2.,fBlank
et_Z/2);

  G4ThreeVector transWallHole(transWallHole_X,transWallHole_Y,0);
  fSolidWall = new G4SubtractionSolid("fSolidWall",
                                      new G4Box("Wall",fWall_X/2.,fWall_
Y/2.,fWall_Z/2.),
                                      new G4Box("WallHole",fWallHole_X/2.
,fWallHole_Y/2.,fWallHole_Z/2.),
                                      myRot,transWallHole);

  fLogicBlanket = new G4LogicalVolume(fSolidBlanket,blanketMat,"fLogicThermalBlanket
");
  fLogicWall = new G4LogicalVolume(fSolidWall,wallMat,"fLogicWall");

  fPhysiBlanket = new G4PVPlacement(0,
                                    G4ThreeVector(0,0,fPhysiBlanket_Z),
                                    "SatelliteThermalBlanket",
                                    fLogicBlanket,
                                    motherVolume,
```

```
                                    false,0,true);

  fPhysiWall = new G4PVPlacement(0,
                                 G4ThreeVector(fPhysiWall_X,fPhysiWall_Y,fPhysiW
all_Z),
                                 "SatelliteWall",
                                 fLogicWall,
                                 motherVolume,
                                 false,0,true);


  //Visualization Attribute

  G4VisAttributes* attGray = new G4VisAttributes(G4Colour::Gray());
  attGray->SetVisibility(true);
  attGray->SetForceAuxEdgeVisible(true);
  fLogicWall->SetVisAttributes(attGray);

  G4VisAttributes* attYellow = new G4VisAttributes(G4Colour::Yellow());
  attYellow->SetVisibility(true);
  attYellow->SetForceAuxEdgeVisible(true);
  fLogicBlanket->SetVisAttributes(attYellow);


}
```