

OpenGeoSys

Open-source Multi-physics Scientific Project

Borehole Heat Exchanger (BHE) Simulation

Getting Started



What is OpenGeoSys

OpenGeoSys is a powerful and versatile open-source project for the simulation of thermo-hydro-mechanical-chemical (THMC) processes in porous and fractured media. The OpenGeoSys is now evolved into OpenGeoSys-6 and always willing to improve the numerical methods by the development from users all over the world. Currently, OGS has been successfully applied in the geotechnical applications, contaminant hydrology, water sources and waste management, geothermal energy and energy storage systems.

About this Tutorial

Borehole heat exchanger plays an important role in geothermal energy exploration. Besides the rule of thumb, there are more and more technicians tend to use simulation tool in the project evaluation and design. This manual is presented to give a tutorial in simulating the performance of BHE through OGS-6. By using this tutorial, anyone could implement the simulation of BHEs considering variation of geothermal properties, borehole characteristics and even layout of multi-boreholes. Hope it will help you in coping with the problem of BHE in application and operation.

Contributors

Wanlong Cai Haibing Shao Boyan Meng Chaofan Chen
Renchao Lu Shuang Chen

Where to find more information about OpenGeoSys

To find more information, document and benchmark of OpenGeoSys, please see the official website of OGS: <https://www.opengeosys.org>.

Contents

1. Quick Start

1.1 Download and installation of OGS

1.2 Get benchmarks

1.3 Running

2. Mathematical Framework

3. Illustration of the Input Files

3.1 Introduction of several Input Files

3.2 The set of GML File

3.3 The set of VTU File

3.4 The set of PRJ File

3.5 Make your own BHE simulation

4. Advance Development Procedure

4.1 Prepare the environment

4.2 Get help with Git

4.3 Compile the original code and build your OGS

4.4 Ready to do some coding

5. Reference

1. Quick Start

In this section, you will learn how to download the latest release of OpenGeoSys and run simulation for the first time. It could be just called a “Quick Start” because in next section we will officially move into the BHE problem.

1.1 *Download and installation of OGS*

You can go to the OpenGeoSys website to find the latest release of OGS platform which could be seen in the user guide on the official website¹ or <https://github.com/ufz/ogs>. Make sure that you choose the correct file in serving your operation system. In this tutorial, all the operation and examples are based on Windows system (x64).

For installation of OGS, you do not need to execute any other set-up. What you should do is put the file downloaded from official website into anywhere you like. Also, you could create a PATH for OGS in your operation system in order to be more convenience in executing program. You could find the PATH option in your System Properties – Environmental Variables – PATH.

Also, we will introduce some advance development skill in section 3. We could get the OGS original code and compile our own executive file to do the simulation.

1.2 *Get benchmarks*

Just like the OGS platform, you should get the benchmarks from the user guide on the OpenGeoSys website¹ (Docs – User Guide – Basics – Introduction – Download benchmarks). You could put the benchmarks in anywhere you want. Also, we will introduce the method of using Git to get the up-to-date benchmarks in Section 3.2. The benchmarks in BHE simulation currently contains several project. To see details about all the benchmarks, please check the ‘Docs-Benchmarks’ in OGS website².

¹ <https://www.opengeosys.org/docs/userguide/basics/introduction>

² <https://www.opengeosys.org/docs/benchmarks>

1.3 Running

Although you may find the `ogs.exe` in the OGS-bin folder. But it could not be operated by click. For operating the simulation, you need three input files. It will be illustrated in the next section. And in this section, we just need to find the `.gml`, `.vtu` and `.prj` files in your benchmarks folder (The path is your benchmarks folder-Parabolic-T). If you get the benchmarks using Git, the path of these files is your clone folder-Tests-Data-Parabolic-T.

There are two way in starting this simulation platform.

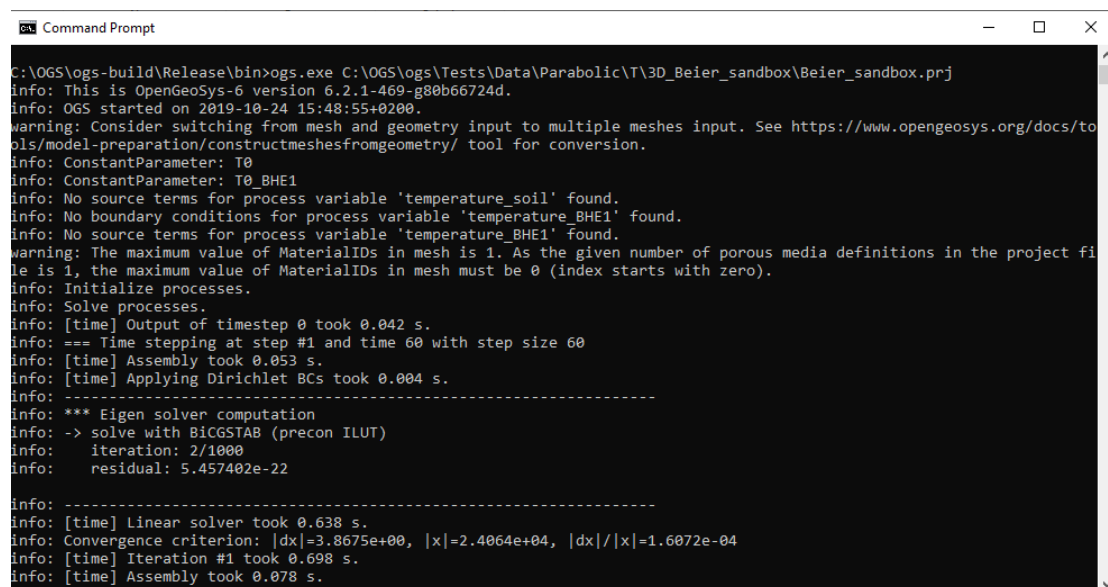
You could use the Command tool in the Windows system by typing following order.

Listing 1.1

The address you put you `ogs.exe/ogs.exe` The address of the project you would like to simulate /project name. `prj`

Press the Enter and you will get the simulation logs in your command window.

And also the simulation results will be in the same file of the OGS. exe.



```
C:\OGS\ogs-build\Release\bin>ogs.exe C:\OGS\ogs\Tests\Data\Parabolic-T\3D_Beier_sandbox\Beier_sandbox.prj
info: This is OpenGeoSys-6 version 6.2.1-469-g80b66724d.
info: OGS started on 2019-10-24 15:48:55+0200.
warning: Consider switching from mesh and geometry input to multiple meshes input. See https://www.opengeosys.org/docs/tools/model-preparation/constructmeshesfromgeometry/ tool for conversion.
info: ConstantParameter: T0
info: ConstantParameter: T0_BHE1
info: No source terms for process variable 'temperature_soil' found.
info: No boundary conditions for process variable 'temperature_BHE1' found.
info: No source terms for process variable 'temperature_BHE1' found.
warning: The maximum value of MaterialIDs in mesh is 1. As the given number of porous media definitions in the project file is 1, the maximum value of MaterialIDs in mesh must be 0 (index starts with zero).
info: Initialize processes.
info: Solve processes.
info: [time] Output of timestep 0 took 0.042 s.
info: == Time stepping at step #1 and time 60 with step size 60
info: [time] Assembly took 0.053 s.
info: [time] Applying Dirichlet BCs took 0.004 s.
info: -----
info: *** Eigen solver computation
info: -> solve with BiCGSTAB (precon ILUT)
info: iteration: 2/1000
info: residual: 5.457402e-22
info: -----
info: [time] Linear solver took 0.638 s.
info: Convergence criterion: |dx|=3.8675e+00, |x|=2.4064e+04, |dx|/|x|=1.6072e-04
info: [time] Iteration #1 took 0.698 s.
info: [time] Assembly took 0.078 s.
```

Figure 1.1 execute ogs.exe in the command window

Also, you could use the `.bat` file to facilitate your operation. You could build a `.bat` file in any folder of your computer and typing following order.

Listing 1.2

The address you put you ogs.exe/ogs.exe The address of the project you would like to simulate /project name. prj > results.txt

In this way, you will soon have a results file which notes all the logs in your operation. Also the simulation results will appear in the same file with the .bat file. (To edit the .bat file, Notepad++³ is recommended to use.)

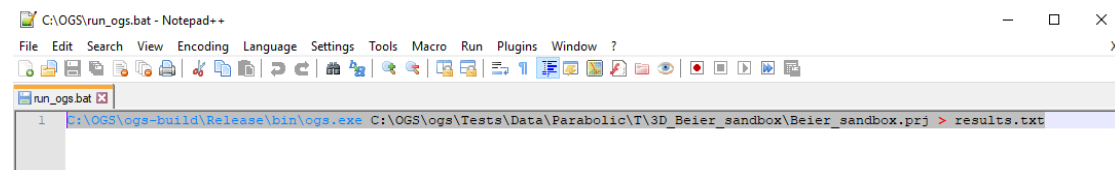


Figure 1.2 execute ogs.exe with the help of .bat file

Because of the inconvenience of typing the address of benchmarks, we recommend that the .bat file should be put in the same file of input files. Then the order in the .bat file could be simplified as following.

Listing 1.3

The address you put you ogs.exe/ogs.exe The project name you would like to simulate. prj > results.txt

1.4 Observing the results

Since we have finished the calculation of our project, the last mission is to observe our simulation results. In OGS package we have a Data Explorer tool⁴ which could visualize our simulation results. We also prefer another multi-platform data analysis and visualization application, ParaView⁵. But the Data Explorer tool is crucial in generating the mesh (in Section 2.3) so it need to be familiar.

³ <https://notepad-plus-plus.org/>

⁴ <https://www.opengeosys.org/docs/DataExplorerManual>

⁵ <https://www.paraview.org/>

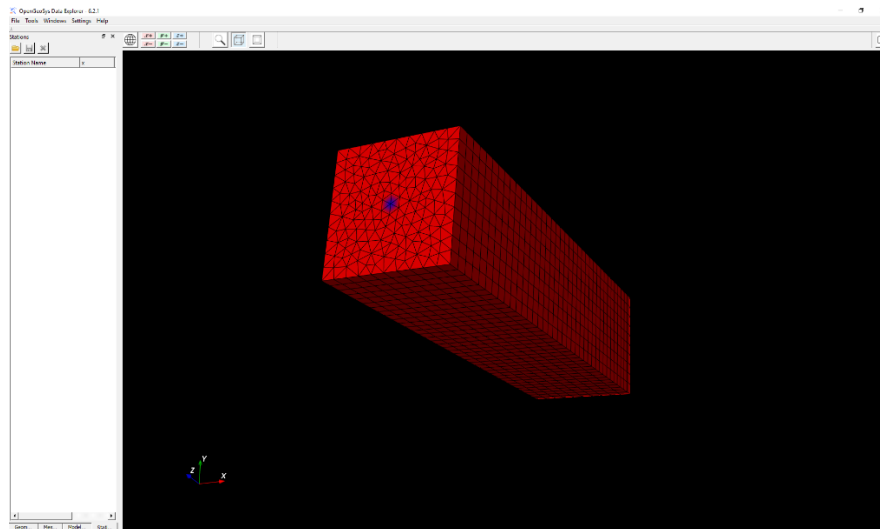


Figure 1.3 Visualize the results in Data explorer

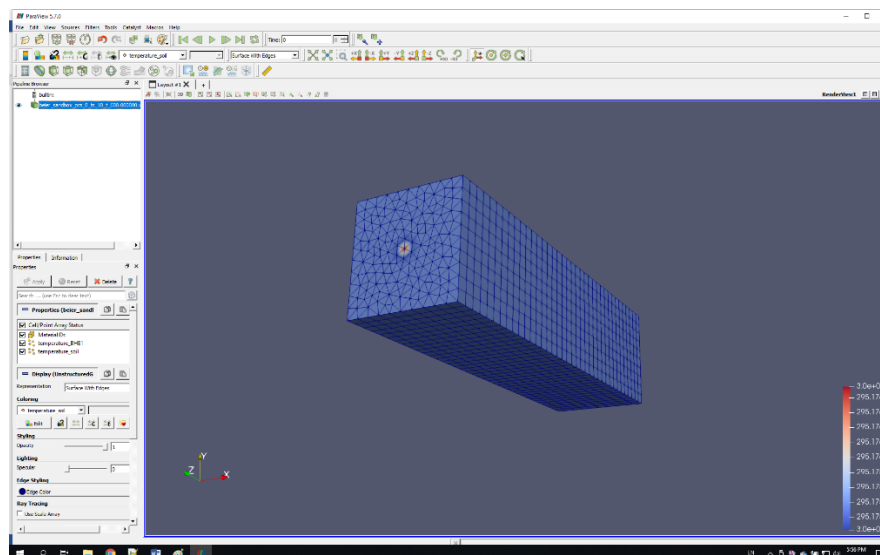


Figure 1.4 Visualize the results in ParaView

In ParaView, you could use the Spreadsheet View to see the simulation data of each point and export data to do specific analysis.

2. Mathematical Framework

This part aims to give an explanation of the mathematical framework in configuring the Heat_Transport_BHE process provided in OpenGeoSys. The numerical method implemented in OGS-6 is the so-called double-continuum finite element method ('DC-FEM'). This approach was originally proposed by Al-Khoury et al. (2010) and extended by Diersch et al. (2011a; 2011b). It was then implemented in OpenGeoSys by Shao et al. (2016).

This modelling approach has the following assumptions:

- *The subsurface is considered to be a 3D continuum, while the BHE is represented by 1D line elements as the second continuum.*
- *The heat transfer between different BHE components is simulated by the Capacity-Resistance-Model (CaRM) in analogy to the electrical circuits.*
- *In the subsurface continuum, both heat convection and heat conduction are governed by the thermal energy conservation equation.*

In the borehole continuum, each pipe is assigned with one governing equation, with the thermal convection in the pipeline simulated. Also, for each grout zone surrounding the pipeline, the thermal conduction equation was simulated. For details of the coupling between different borehole components and continua, interested readers may refer to Diersch et al. (2011a; 2011b).

3. Illustration of the Input Files

In this section, we will focus on the three input files for OGS. You will learn how to create each file and customize your research problem. In this way, you could implement the whole procedure of OpenGeoSys platform and also carry out your own simulation. You could also check the documents for all the input files in Doxygen website⁶ for OGS.

3.1 Introduction of several Input Files

Because of simplification of input files in OGS-6, we only need three type of files to execute the simulation which including .gml file, .vtu file and .prj file. After preparing these files, all we need is put them in one folder and set up the OGS platform using the method in Section 1.

3.2 The set of GML File

The gml file is used to define the geometries characteristics which contain points, polylines and triangulated surfaces. These geometries features will be used in project files (e.g. in definition of boundary conditions).

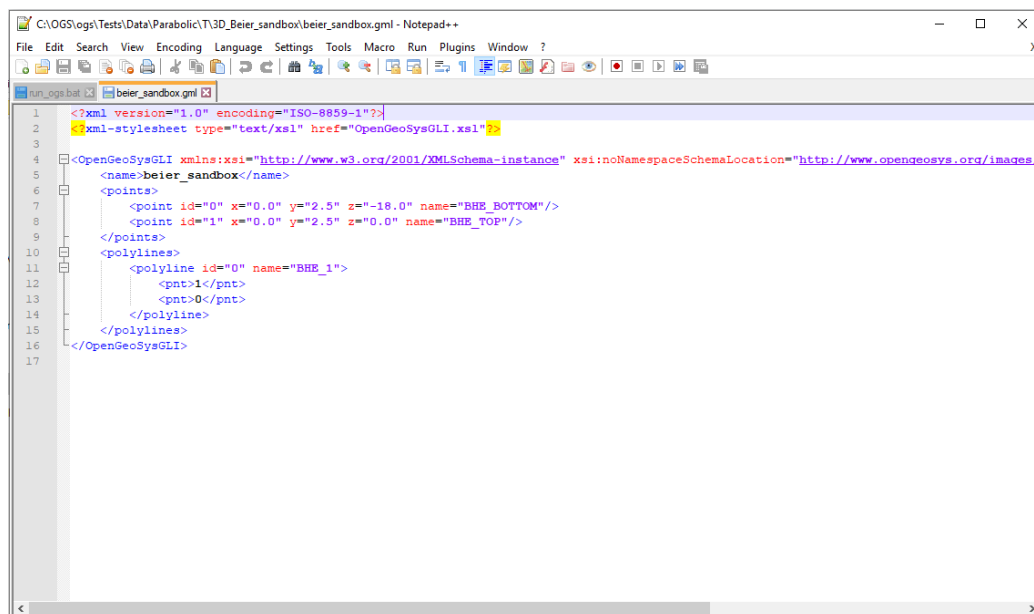


Figure 3.1 The set of .gml file

In the 'name' tag, you could just write the name of your project.

⁶ https://doxygen.opengeosys.org/ogs_file_param__ProjectFile.html

In the 'point' tag, you should define the point which could be used to be a reference in the definition of surfaces and polylines. The definition needs to set the id, name and coordinates. The following is an example.

Listing 3.1

```
<points>  
  <point id="0" x="0.0" y="2.5" z="-18.0" name="BHE_BOTTOM"/>  
  <point id="1" x="0.0" y="2.5" z="0.0" name="BHE_TOP"/>  
</points>
```

In the 'polylines' tag, the definition consists of the id and name of polylines. Also, the points ID which is linked to your polylines.

Listing 3.2

```
<polylines>  
  <polyline id="0" name="BHE_1">  
    <pnt>1</pnt>  
    <pnt>0</pnt>  
  </polyline>  
</polylines>
```

Despite the id and name, you should define three points of the single element of your surface in the 'surfaces' tag using the point's id defined in 'point' tag.

Listing 3.3

```
<surfaces>  
  <surface id="0" name="left">  
    <element p1="0" p2="1" p3="2"/>  
    <element p1="0" p2="3" p3="2"/>  
  </surface>  
</surfaces>
```

3.3 The set of VTU File

The vtu file is actually the calculating mesh for your problem. You could view this file in Data Explorer or ParaView. To create your working mesh, especially

for the BHE simulation project, you could download a handy meshing tool developed by PhD of UFZ (Thanks to Philipp Hein and Chaofan Chen!). By using this tool, we could easily generate our mesh file. Also, you could use the Gmsh software to draw the mesh by yourself (It could be a little bit different for green hand in BHE simulation).

After we get the meshing tool⁷ (You also can build it by yourself, the code could be downloaded at https://github.com/ChaofanChen/meshing_tool_BHE. The way to build this exe file could be seen in Section 3.), we could implement the set of vtu file conveniently. You can look up the documentation for the detail of this meshing tool at https://github.com/WanlongCai/meshing_tool_BHE.

First of all, we should get the Gmsh software which is useful in our generating procedure⁸. So we are already having the 'BHE_meshing_tool.exe' (It should be connected with '_'), 'gmsh.exe', 'example.inp' (It could be get from the original code or create by yourself). Just create a bat file within following information.

Listing 3.4

```
BHE_meshing_tool.exe example.inp > log.txt
```

```
del example.msh
```

```
ren example.bhe.msh example.msh
```

Then we edify the content of the inp file. We need to set the width, length and depth of simulation domain. The box properties define the refining area. Also we could set the element size of domain (Element size at the boundary of the refinement box and the outer boundaries), layer characteristics (material group; number of elements in this layer; thickness of elements in this layer) and BHE properties (Make sure we should define the top and bottom of BHE which means the BHE could be shorter than the domain).

Listing 3.5

⁷ A website

⁸ <http://gmsh.info/>

WIDTH 300

LENGTH 300

DEPTH 100

BOX 100 100 100

ELEM_SIZE 5 30

LAYER 0 8 0.5

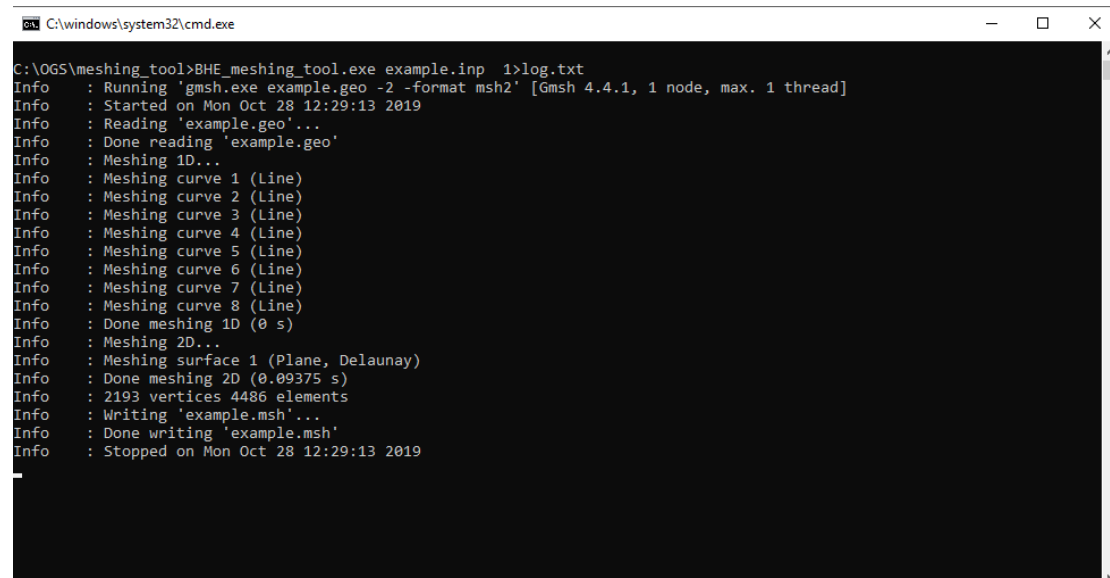
LAYER 1 48 2

// BHE# x y z_top z_bottom radius

BHE 0 -6 144 -4 -54 0.075

BHE 1 -6 150 -4 -54 0.075

Finally, we run the bat file and the mesh could be obtained easily.



```
C:\windows\system32\cmd.exe
C:\OGS\meshing_tool>BHE_meshing_tool.exe example.inp 1>log.txt
Info : Running 'gmsh.exe example.geo -2 -format msh2' [Gmsh 4.4.1, 1 node, max. 1 thread]
Info : Started on Mon Oct 28 12:29:13 2019
Info : Reading 'example.geo'...
Info : Done reading 'example.geo'
Info : Meshing 1D...
Info : Meshing curve 1 (Line)
Info : Meshing curve 2 (Line)
Info : Meshing curve 3 (Line)
Info : Meshing curve 4 (Line)
Info : Meshing curve 5 (Line)
Info : Meshing curve 6 (Line)
Info : Meshing curve 7 (Line)
Info : Meshing curve 8 (Line)
Info : Done meshing 1D (0 s)
Info : Meshing 2D...
Info : Meshing surface 1 (Plane, Delaunay)
Info : Done meshing 2D (0.09375 s)
Info : 2193 vertices 4486 elements
Info : Writing 'example.msh'...
Info : Done writing 'example.msh'
Info : Stopped on Mon Oct 28 12:29:13 2019
```

Figure 3.2 Generating of mesh

Last but not least, we should open the 'example.msh' by Data Explorer and transfer its format to vtu file (File-export-VTK). Now you have your calculating mesh and you can open it through ParaView.

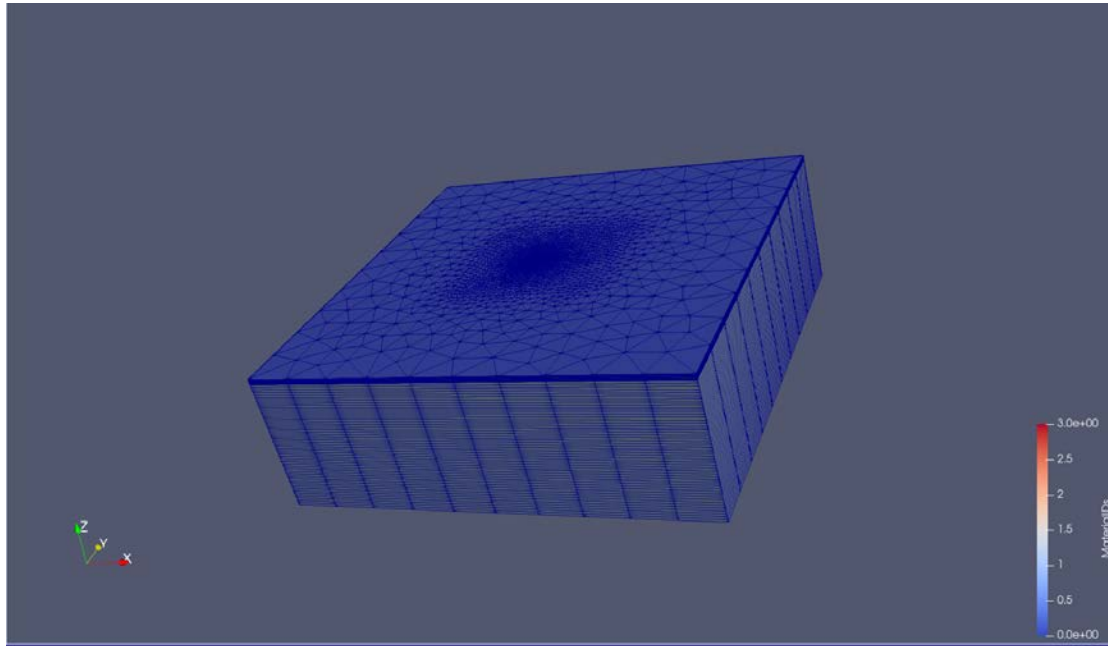


Figure 3.3 The finished calculating mesh

3.4 The set of PRJ File

We have already figure out the gml and vtu file, the last one is the prj file which is the most important one. The detail for the prj file could be seen at https://doxygen.opengeosys.org/ogs_file_param__prj__processes__process__HEAT__TRANSPORT_BHE__borehole_heat_exchangers.html.

There are several parts in prj file. You should know that all the items' definitions in prj file have a beginning and an end just like `<mesh>` and `</mesh>`. Some of the items have their sub-item just like `<processes>` and `<process>`. Please make sure you set up the items correctly and completely.

<mesh>: You should set the name of vtu which will be used in your simulation.

<geometry>: You should set the name of gml which will be used in your simulation.

<processes>

In the configuration of Heat_Transport_BHE process, it is generally configured as follows.

- **<name>**: should be HeatTransportBHE.
- **<type>**: should be HEAT_TRANSPORT_BHE.
- **<integration_order>**: It is the order of the integration method for element-

wise integration, normally set to 2.

- `<process_variables>`: The variables of the HEAT_TRANSPORT_BHE process are temperature_soil and temperature_BHE1. For multiple boreholes, the name temperature_BHE2, temperature_BHE3, temperature_BHE4, etc can be added.

The `<borehole_heat_exchangers>` contains a lot of important properties in our simulation. In case of this part of definition is crucial in the whole BHE simulation, we will explain all the details in this files.

Keyword `<borehole>`

The borehole `<length>` and `<diameter>` are defined here.

Keyword `<type>`

Currently there are 4 types of BHE available. Following the convention in Diersch et al. (2011a), they are named as 1U , 2U , CXA and CXC types.

- 1U, means there is only one single U-tube installed in the borehole;
- 2U, double U-tubes installed in the borehole;
- CXA, coaxial pipe with annular space as the inlet downwards flow and the centre part as outlet upwards flow;
- CXC, coaxial pipe with a reversed flow direction to CXA type.

The cross-sections of these 4 types of BHEs are illustrated in the following figures.

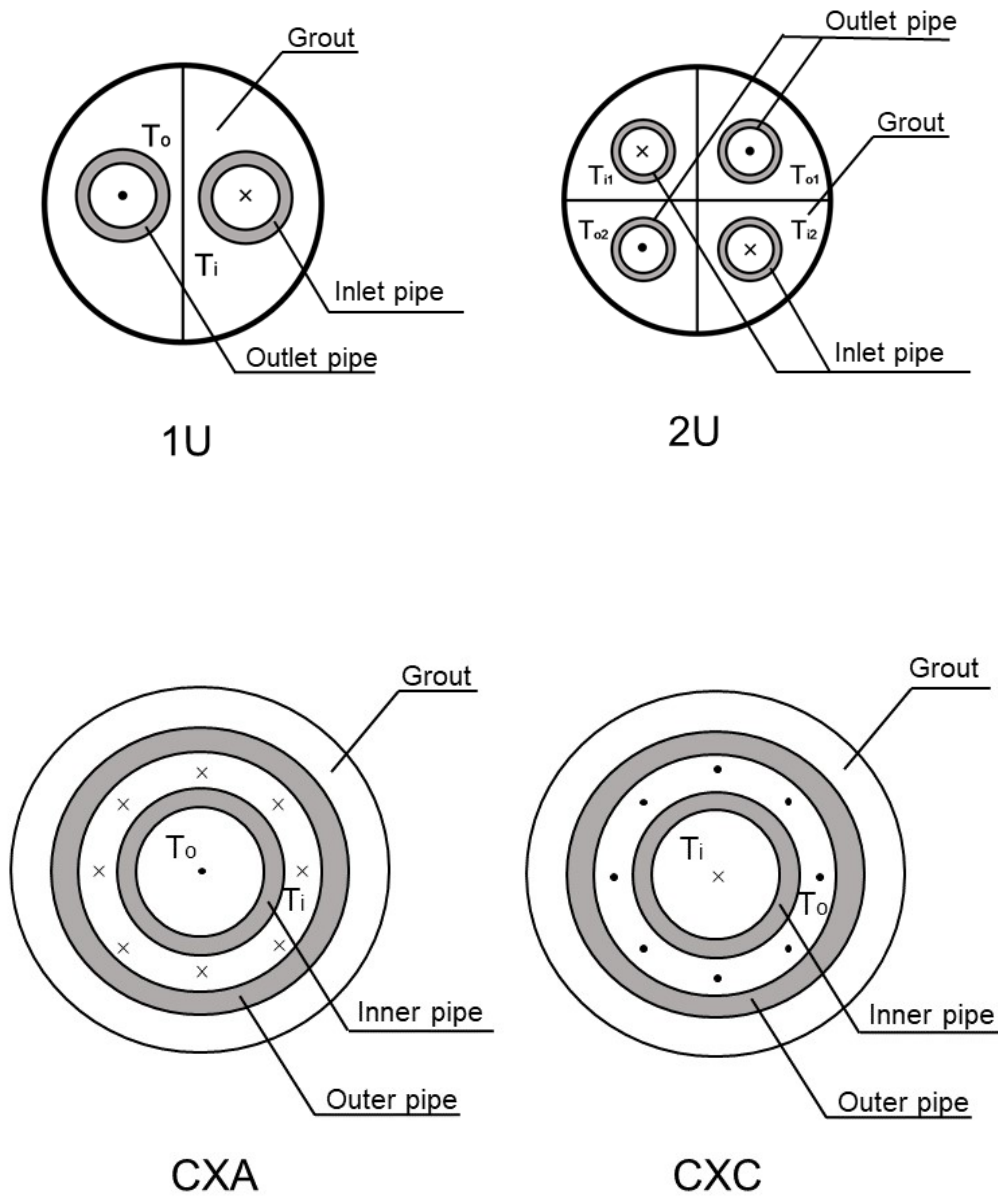


Figure 3.4 Diagram for 4 BHE types

Keyword <pipes>

The properties of the pipes are defined in this section. For different types of BHE, the pipes are also configured differently.

- For coaxial pipes (CXA or CXC), <longitudinal_dispersion_length> should be given.
- For 1U and 2U type pipe, the <distance_between_pipes> must be given, along side with the <longitudinal_dispersion_length>.

Keyword <flow_and_temperature_control>

Four type of flow and temperature control patterns are provided in OGS.

- FixedPowerConstantFlow:

It means the BHE has a fixed thermal load and the refrigerant flow rate in the borehole is kept as a constant. The fixed heating load value was defined by the key word <power> , and the flow rate is given by <flow_rate>.

- FixedPowerFlowCurve:

It means the BHE has a fixed thermal load and the flow rate is following a time dependent curve. The key word <power> is kept the same, while the flow rate is defined by a curve in the <curves>.

- PowerCurveConstantFlow:

It means BHE has a constant flow rate while the power is various following a curve. The key word <flow_rate> applies here, along with the curve defined in the <curves>.

- TemperatureCurveConstantFlow:

It means BHE has a constant <flow_rate> while the inflow temperature following the values defined in the <curves>.

For all the flow and temperature control options, OpenGeoSys calculates the inlet temperature of each BHE internally. For each BHE, temperature on its inlet pipe is always set as a Dirichlet type boundary condition. Depending on the choice of <flow_and_temperature_control>, the inflow temperature will be calculated dynamically in each time step and iteration to satisfy the given constrains.

All the parameters follow the SI standard.

Keyword <grout>

The thermal properties of the grout material is defined here.

Keyword <refrigerant>

The thermal properties of the circulating fluid is defined here.

<media>

In the definition of this item, we need to set up the media involving in the

simulation. The media ID is used to distinguish different media based on the Material IDs. The type of the phase could be Liquid, Gas, Solid and so on (Some of the phase we do not use in BHE simulation). In every type you need to set the properties including heat capacity, thermal conductivity and density. Also, if you have some other parameters needed to be set, you could add them in the <properties> under the <media> item.

<time_loop>

In this item, we could define some detail in calculating process. You should firstly set the nonlinear_solver type which consist of basic_picard or newton (basically we use picard iteration method, but you could try the newton method for strongly nonlinear problem). In the <convergence_criterion>, you should set the <reltol> which means the relative tolerance. In the method of discretization, we always choose the backward Euler method. For the <time_stepping>, you need to set the initial and end time of the simulation. Also, the time step should be defined. The repeat is the iteration time of every single time step, <delta_t>. You are allowed to set different time step for different simulation stage. The following set is to show that the first 100 times iteration have a time step of 60s and the last iteration will maintain a time step of 600s (The simulation will automatically execute the last definition of time step to the end).

Listing 3.6

```
<timesteps>
  <pair>
    <repeat>100</repeat>
    <delta_t>60</delta_t>
  </pair>
  <pair>
    <repeat>1</repeat>
    <delta_t>600</delta_t>
  </pair>
</timesteps>
```

Or, you could set the time step in order to meet the whole simulation time just like Listing 2.7.

Listing 3.7

```
<time_stepping>
  <type>FixedTimeStepping</type>
  <t_initial> 0.0 </t_initial>
  <t_end> 186420 </t_end>
  <timesteps>
    <pair>
      <repeat>3107</repeat>
      <delta_t>60</delta_t>
    </pair>
  </timesteps>
</time_stepping>
```

For the output, you should define the prefix and type of output file and time step for output. In `<timesteps>`, you need to set the `<repeat>` which is the times you run `<each_steps>`. The `<each_steps>` is the interval that you do output between two simulation time. Listing 2.8 is an example. For the full simulation, we have to repeat 3107 times. The first 10 times will execute output for each time. And the last time will execute output for every 1000 times.

Listing 3.8

```
<timesteps>
  <pair>
    <repeat>10</repeat>
    <each_steps>1</each_steps>
  </pair>
  <pair>
    <repeat>1</repeat>
    <each_steps>1000</each_steps>
  </pair>
```

</timesteps>






















Name	Date modified	Type	Size
 beier_sandbox.gml	19/10/21 4:29 PM	GML File	1 KB
 beier_sandbox.prj	19/11/01 11:45 AM	PRJ File	76 KB
 beier_sandbox.vtu	19/10/21 4:29 PM	VTU File	668 KB
 beier_sandbox_pcs_0.pvd	19/11/01 2:05 PM	PVD File	2 KB
 beier_sandbox_pcs_0_ts_0_t_0.000000.vtu	19/11/01 11:45 AM	VTU File	211 KB
 beier_sandbox_pcs_0_ts_1_t_60.000000.vtu	19/11/01 11:46 AM	VTU File	237 KB
 beier_sandbox_pcs_0_ts_2_t_120.000000.vtu	19/11/01 11:46 AM	VTU File	238 KB
 beier_sandbox_pcs_0_ts_3_t_180.000000.vtu	19/11/01 11:46 AM	VTU File	240 KB
 beier_sandbox_pcs_0_ts_4_t_240.000000.vtu	19/11/01 11:46 AM	VTU File	240 KB
 beier_sandbox_pcs_0_ts_5_t_300.000000.vtu	19/11/01 11:46 AM	VTU File	241 KB
 beier_sandbox_pcs_0_ts_6_t_360.000000.vtu	19/11/01 11:46 AM	VTU File	242 KB
 beier_sandbox_pcs_0_ts_7_t_420.000000.vtu	19/11/01 11:46 AM	VTU File	242 KB
 beier_sandbox_pcs_0_ts_8_t_480.000000.vtu	19/11/01 11:46 AM	VTU File	242 KB
 beier_sandbox_pcs_0_ts_9_t_540.000000.vtu	19/11/01 11:46 AM	VTU File	243 KB
 beier_sandbox_pcs_0_ts_10_t_600.000000.vtu	19/11/01 11:46 AM	VTU File	243 KB
 beier_sandbox_pcs_0_ts_1010_t_60600.000000.vtu	19/11/01 12:29 PM	VTU File	252 KB
 beier_sandbox_pcs_0_ts_2010_t_120600.000000.vtu	19/11/01 1:18 PM	VTU File	253 KB
 beier_sandbox_pcs_0_ts_3010_t_180600.000000.vtu	19/11/01 2:02 PM	VTU File	254 KB
 beier_sandbox_pcs_0_ts_3107_t_186420.000000.vtu	19/11/01 2:05 PM	VTU File	254 KB
 results.txt	19/11/01 2:05 PM	Text Document	5,578 KB
 run_ogs.bat	19/11/01 11:41 AM	Windows Batch File	1 KB

Figure 3.5 The output results

<parameters>

In this item, we need to define some parameters involving in our simulation. The name, type and values should be set.

<process_variables>

In this item, we should set the name, components, order and initial condition of process variables which is defined in the first <processes>. Also, you are allowed to add boundary conditions or source terms here⁹.

<nonlinear_solvers> & <linear_solvers>

The solver of OGS need to be set in this items. The further detail could be seen in the Doxygen document. Usually we need to make some adjustments with maximum iteration in <max_iter>.

<curves>

You could define the values of the variable you set in other item. The coordinates and values should be entered (You can use Excel or other tools to

⁹ https://doxygen.opengeosys.org/ogs_ctest_prj__Parabolic__T__2D_BHE_array__bhe2d__prj.html

help you do the data processing).

3.5 *Make your own BHE simulation*

Now you have already known the simulation procedure of BHE. In conclusion, you should prepare the official release OGS package and other software first, such as ParaView, Notepad++. Then you need to set up all the input files including gml, vtu and prj file which are matched with actual situation of your project. Last but not least, you just execute the OGS platform by bat file or command window and wait for the simulation results. The function of OGS is enough to lots of simulation scenario, but if you need advance function or self-development. You should try to dive into the code and build your own OGS. The section 3 will tell you about that.

4. Advance Development Procedure

In Section 1 and 2, we have already known how to implement our own simulation for BHE. But in that way we could only use the official release version of OGS. The OGS's functionality is evolving and expanding everyday by the work of UFZ and developers all around the world. Thus it is necessary to keep up with the up-to-date version of OGS. So we will learn how to build your own OGS and be an OGS developer¹⁰.

4.1 *Prepare the environment*

OGS is based on C++ so that we should prepare the environment in making convenience of compiling the code.

At the beginning, we should install a compiler. We prefer to choose the Visual Studio 2015 and up. The free Community Edition of Visual Studio is a perfect choice. You could download and install the Visual Studio Community¹¹. During installation, please select the workload 'Desktop Development with C++'.

Secondly, we should install Python 3 which could also be used in the coding. Just make sure you add Python 3.X to PATH, install for all users and have *pip* enabled in the installation. Then we need to install Conan package manager which could help us install all required libraries conveniently. With the assistance of *pip*, we can get Conan easily. Type the following command in the command window.

Listing 4.1

```
pip3 install --user conan
```

To check if conan is being installed properly, type this command and the version number of conan will display on the command window.

Listing 4.2

```
conan --version
```

¹⁰ <https://www.opengeosys.org/docs/devguide/getting-started/prerequisites/>

¹¹ <https://visualstudio.microsoft.com/>

Finally, we should install CMake¹². CMake is an open-source, cross-platform family of tools designed to build, test and package software. When you execute the install, please make sure that ‘Add CMake to the system path for all users’ option is on.

4.2 Get help with Git

Git is a powerful and distributed version control system which could help us manage the code. Because the source code of OGS is hosted on GitHub, we need some version control tool to facilitate our code developing. You could get the ‘Git’ program or we prefer to use the GitKraken which is a free Git GUI client. You can choose one you would like to use.

Also, we could easily get the source code of OGS by using Git. After installing the Git/GitKraken, you could use the ‘clone’ to download the OGS source code. Just set the path for your repository and the source code. You could get the latest OGS code and benchmarks conveniently.

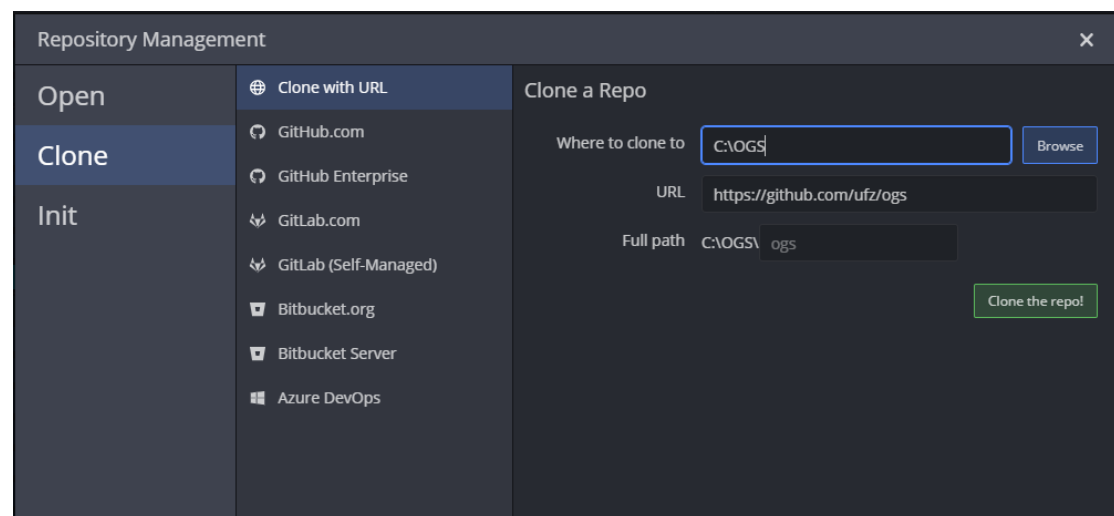


Figure 4.1 Clone your repository

4.3 Compile the original code and build your OGS

Once you get the source code of OGS, you could do the code compiling. Use CMake to compile the original code.

¹² <https://cmake.org/download/>

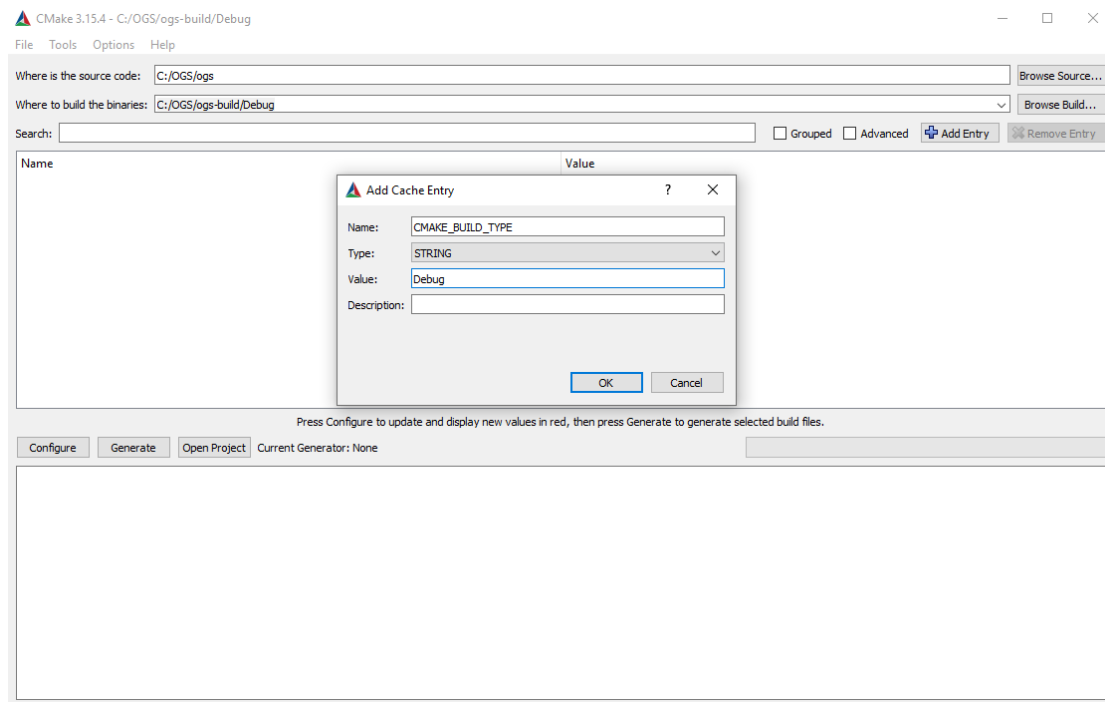


Figure 4.2 Configure your code and generate your project

Please note that the build path could be any folder but we prefer to choose the same folder with the source code. Also, you should add an entry before configuration which could guarantee you get the correct pattern of your project (Release pattern is for publish and debug pattern will contain more details in compiling).

After finishing the generation, you should open this project (or the sln file) by Visual Studio. Then set the ogs as the startup project.

Finally, click the button to build the solution. Now you get your own ogs.exe.

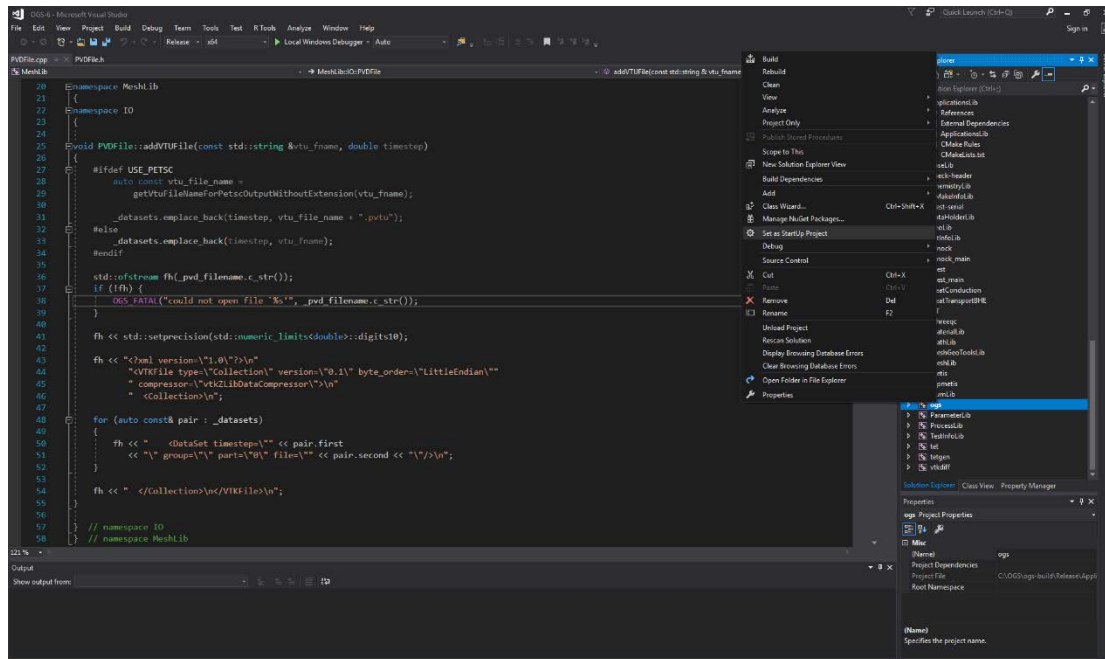


Figure 4.3 Build the solution and get the ogs.exe

4.4 Ready to do some coding

You could use Git/GitKraken to do the code management and GitHub to view or share your code. If you develop some new function or library for OGS, you can just pull request on GitHub and our manager will check your request and determine whether your work could be merged into the official code. Also, we also have the Jerkins testing platform and other operation tools. Please check the website of OGS¹³.

Hope you enjoy this powerful and versatile simulation tool and have fun in doing the BHE simulation.

¹³ <https://www.opengeosys.org/docs/devguide/>

5. References

- [1] Al-Khoury, R., Kölbel, T., Schramedei, R.: Efficient numerical modeling of borehole heat exchangers. *Comput. Geosci.* 36(10), 1301–1315 (2010).
- [2] Diersch, H.-J.G., Bauer, D., Heidemann, W., Rühaak, W., Schätzl, P.: Finite element modeling of borehole heat exchanger systems: part 1. Fundamentals. *Comput. Geosci.* 37(8), 1122–1135 (2011a).
- [3] Diersch, H.-J.G., Bauer, D., Heidemann, W., Rühaak, W., Schätzl, P.: Finite element modeling of borehole heat exchanger systems: part 2. Numerical simulation. *Comput. Geosci.* 37(8), 1136–1147 (2011b).
- [4] Hein, P., Kolditz, O., Görke, U.-J., Bucher, A., Shao, H.: A numerical study on the sustainability and efficiency of borehole heat exchanger coupled ground source heat pump systems. *Appl. Therm. Eng.* 100, 421–433 (2016).
- [5] Shao, Haibing, Philipp Hein, Agnes Sachse, and Olaf Kolditz. *Geoenergy modeling II: shallow geothermal systems*. Springer International Publishing, 2016.