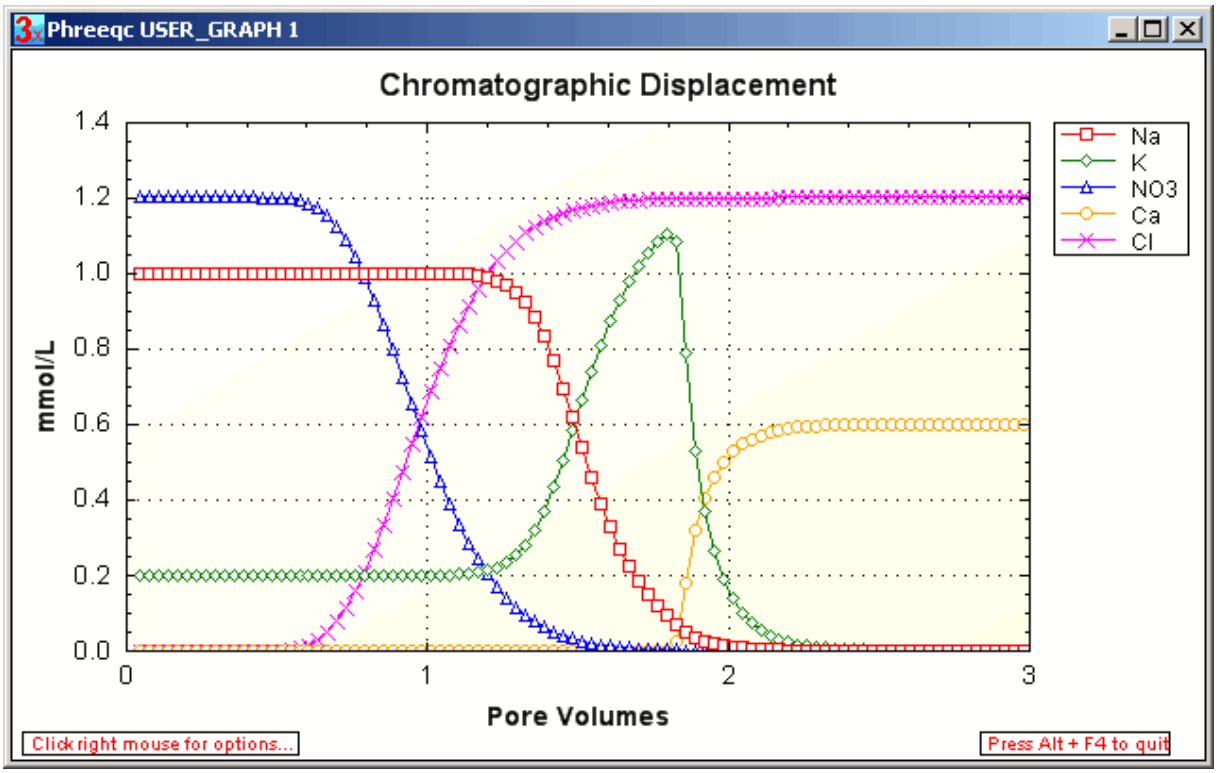
A short tutorial how to implement a PHREEQC 1d transport and ion exchange reaction example in ORCHESTRA

Hans Meeussen 26/11/2024

This document describes how implement a simple PHREEQC ion exchange and 1d transport example in ORCHESTRA. The example was taken from the very useful PHREEQC site by Tony Appelo and the PHREEQC description can be found here:

Chromatographic displacement of ions (hydrochemistry.eu)

The example demonstrates how to calculate the composition of a chromatographic column that is first in equilibrated with 1 mmol $NaNO_3$ plus 0.2 mmol KNO_3 and subsequently flushed with a 0.6 mmol $CaCl_2$ solution. This results in the following concentration breakthrough pattern at the outflow:



The definition of the PHREEQC initial condition of input and column solution is, as always, very compact. We can see here the composition of both solutions, including the 0.0011 mol of exchanger surface present.

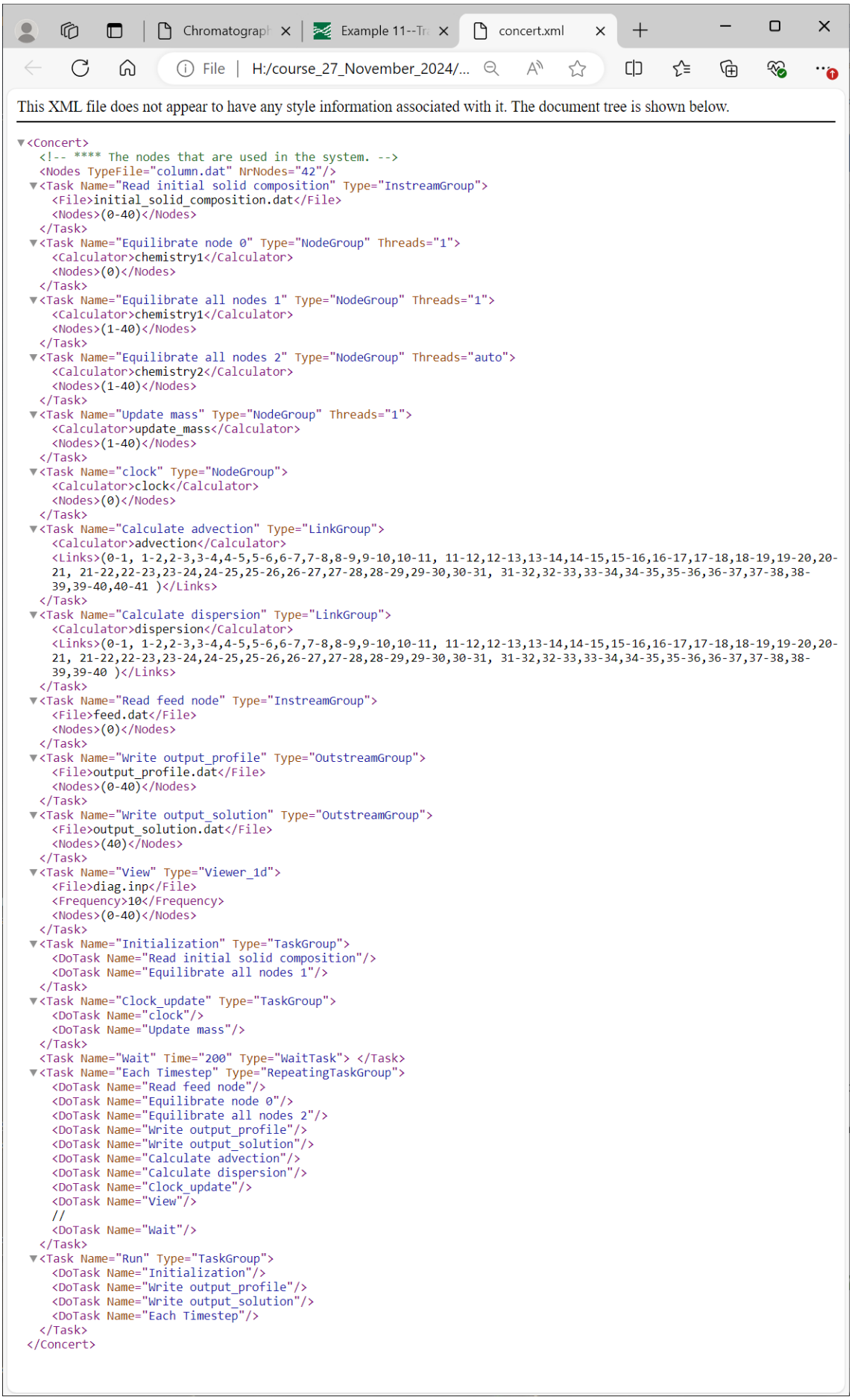
```
TITLE Example 11.--Transport and cation exchange.
SOLUTION 0  CaCl2
        units           mmol/kgw
```

```
        temp            25.0
        pH              7.0     charge
        pe              12.5    O2(g)   -0.68
        Ca              0.6
        Cl              1.2
SOLUTION 1-40  Initial solution for column
        units           mmol/kgw
        temp            25.0
        pH              7.0     charge
        pe              12.5    O2(g)   -0.68
        Na              1.0
        K               0.2
        N(5)            1.2
END
EXCHANGE 1-40
        -equilibrate 1
        X               0.0011
COPY cell 1 101
END
```

This is short description is very elegant, but to reproduce it with another code it is necessary to understand what is going on behind the screen!

ORCHESTRA implementation

The definition of all the required system parts and calculation tasks can be found in the concert.xml file. A screenshot is shown here, but the complete file is present in the accompanying folder.




```
<Concert>
  <!-- **** The nodes that are used in the system. -->
  <Nodes TypeFile="column.dat" NrNodes="42"/>
  <Task Name="Read initial solid composition" Type="InstreamGroup">
    <File>initial_solid_composition.dat</File>
    <Nodes>(0-40)</Nodes>
  </Task>
  <Task Name="Equilibrate node 0" Type="NodeGroup" Threads="1">
    <Calculator>chemistry1</Calculator>
    <Nodes>(0)</Nodes>
  </Task>
  <Task Name="Equilibrate all nodes 1" Type="NodeGroup" Threads="1">
    <Calculator>chemistry1</Calculator>
    <Nodes>(1-40)</Nodes>
  </Task>
  <Task Name="Equilibrate all nodes 2" Type="NodeGroup" Threads="auto">
    <Calculator>chemistry2</Calculator>
    <Nodes>(1-40)</Nodes>
  </Task>
  <Task Name="Update mass" Type="NodeGroup" Threads="1">
    <Calculator>update_mass</Calculator>
    <Nodes>(1-40)</Nodes>
  </Task>
  <Task Name="clock" Type="NodeGroup">
    <Calculator>clock</Calculator>
    <Nodes>(0)</Nodes>
  </Task>
  <Task Name="Calculate advection" Type="LinkGroup">
    <Calculator>advection</Calculator>
    <Links>(0-1, 1-2,2-3,3-4,4-5,5-6,6-7,7-8,8-9,9-10,10-11, 11-12,12-13,13-14,14-15,15-16,16-17,17-18,18-19,19-20,20-
    21, 21-22,22-23,23-24,24-25,25-26,26-27,27-28,28-29,29-30,30-31, 31-32,32-33,33-34,34-35,35-36,36-37,37-38,38-
    39,39-40,40-41 )</Links>
  </Task>
  <Task Name="Calculate dispersion" Type="LinkGroup">
    <Calculator>dispersion</Calculator>
    <Links>(0-1, 1-2,2-3,3-4,4-5,5-6,6-7,7-8,8-9,9-10,10-11, 11-12,12-13,13-14,14-15,15-16,16-17,17-18,18-19,19-20,20-
    21, 21-22,22-23,23-24,24-25,25-26,26-27,27-28,28-29,29-30,30-31, 31-32,32-33,33-34,34-35,35-36,36-37,37-38,38-
    39,39-40 )</Links>
  </Task>
  <Task Name="Read feed node" Type="InstreamGroup">
    <File>feed.dat</File>
    <Nodes>(0)</Nodes>
  </Task>
  <Task Name="Write output_profile" Type="OutstreamGroup">
    <File>output_profile.dat</File>
    <Nodes>(0-40)</Nodes>
  </Task>
  <Task Name="Write output_solution" Type="OutstreamGroup">
    <File>output_solution.dat</File>
    <Nodes>(40)</Nodes>
  </Task>
  <Task Name="View" Type="Viewer_1d">
    <File>diag.inp</File>
    <Frequency>10</Frequency>
    <Nodes>(0-40)</Nodes>
  </Task>
  <Task Name="Initialization" Type="TaskGroup">
    <DoTask Name="Read initial solid composition"/>
    <DoTask Name="Equilibrate all nodes 1"/>
  </Task>
  <Task Name="Clock_update" Type="TaskGroup">
    <DoTask Name="clock"/>
    <DoTask Name="Update mass"/>
  </Task>
  <Task Name="Wait" Time="200" Type="WaitTask"> </Task>
  <Task Name="Each Timestep" Type="RepeatingTaskGroup">
    <DoTask Name="Read feed node"/>
    <DoTask Name="Equilibrate node 0"/>
    <DoTask Name="Equilibrate all nodes 2"/>
    <DoTask Name="Write output_profile"/>
    <DoTask Name="Write output_solution"/>
    <DoTask Name="Calculate advection"/>
    <DoTask Name="Calculate dispersion"/>
    <DoTask Name="Clock_update"/>
    <DoTask Name="View"/>
    //
    <DoTask Name="Wait"/>
  </Task>
  <Task Name="Run" Type="TaskGroup">
    <DoTask Name="Initialization"/>
    <DoTask Name="Write output_profile"/>
    <DoTask Name="Write output_solution"/>
    <DoTask Name="Each Timestep"/>
  </Task>
</Concert>
```

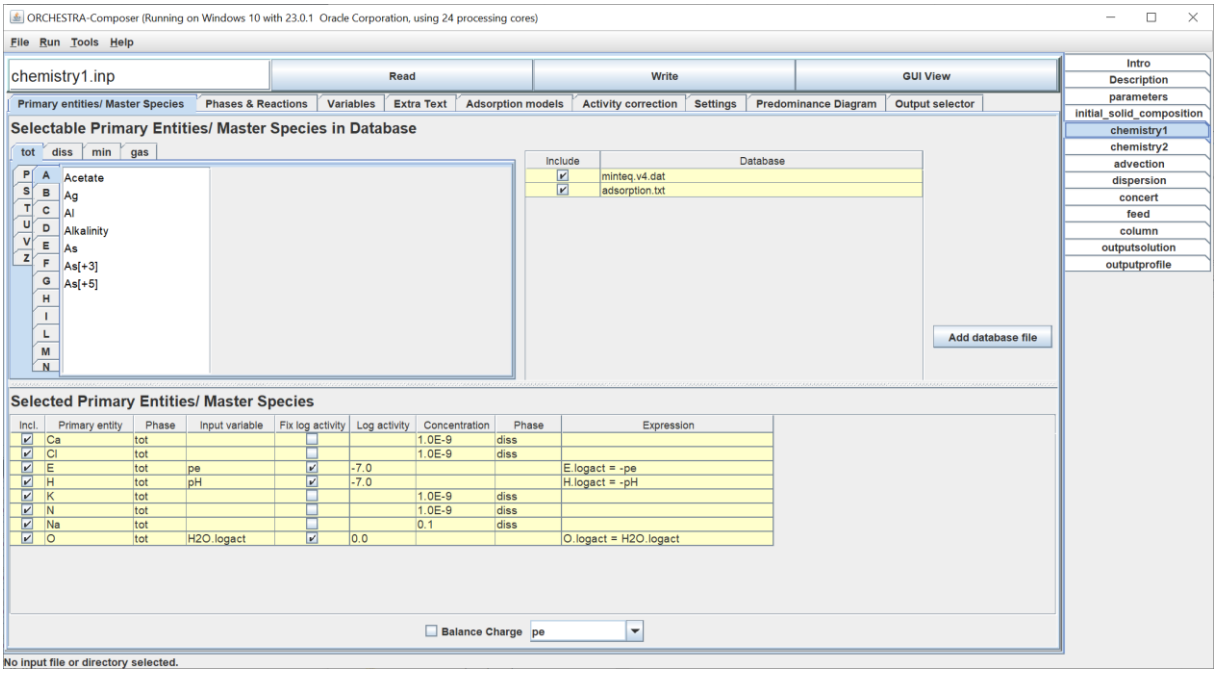
To define the same system in ORCHESTRA, we need the same 40 cells for the transport system (cell1-40), plus a cell that represents the input solution (cell 0) and another one that represents a waste bucket (cell 41) that will be used as a sink at the end of the column. We actually define 42 cells.

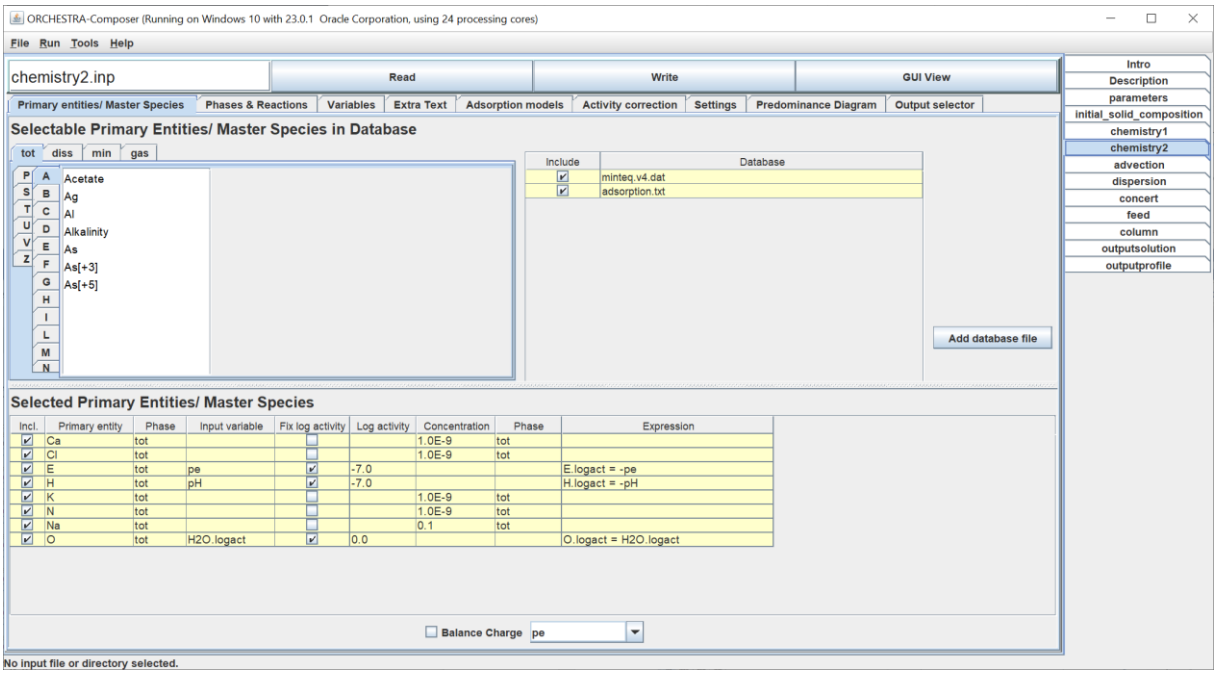
In the concert file a number of different calculation or input output tasks are defined, and at the bottom the “Run” task contains the tasks that are carried out during a run.

Here we can see that the system is first initialized by reading the composition of cells 0-40 from an input file, and subsequently for each of these cells a solver (chemistry1.inp) is used to calculate the initial chemical composition.

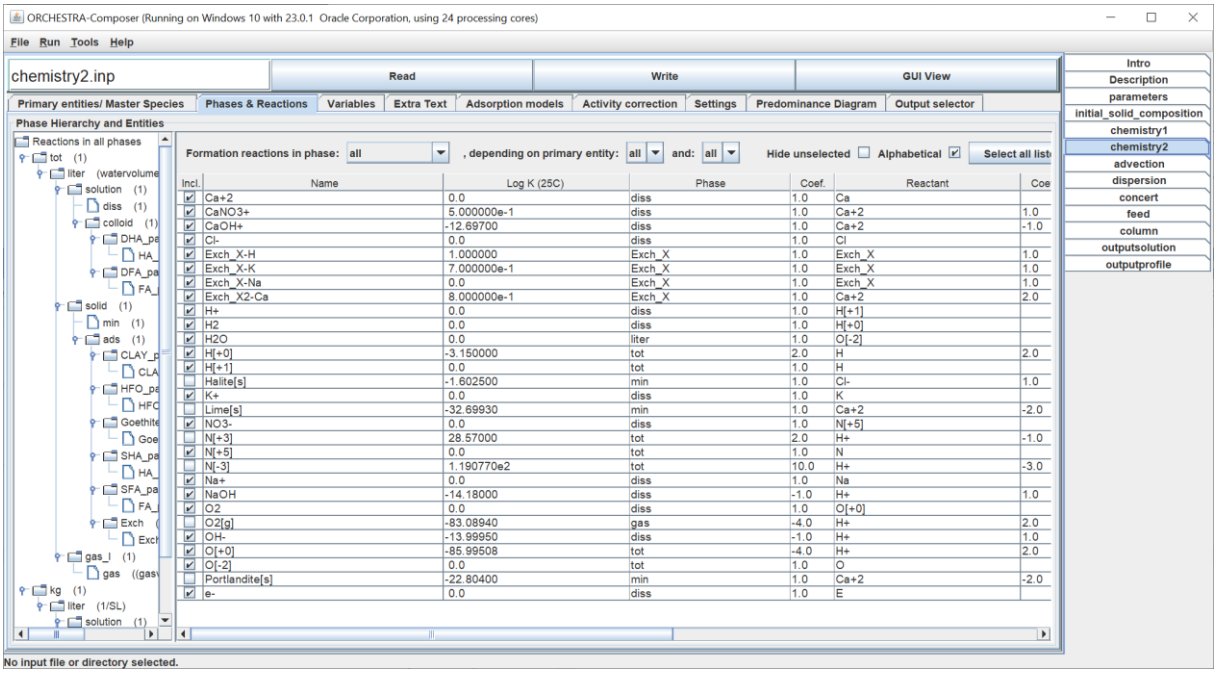
We need two solvers in this system; one to calculate the initial composition of the system with given dissolved concentrations as input (<name>.diss) , and a second calculator that is used during transport that uses the given total amount of substances (<name>.tot) in a cell as input.

So these solvers only differ in the definition of the primary entities / master species:

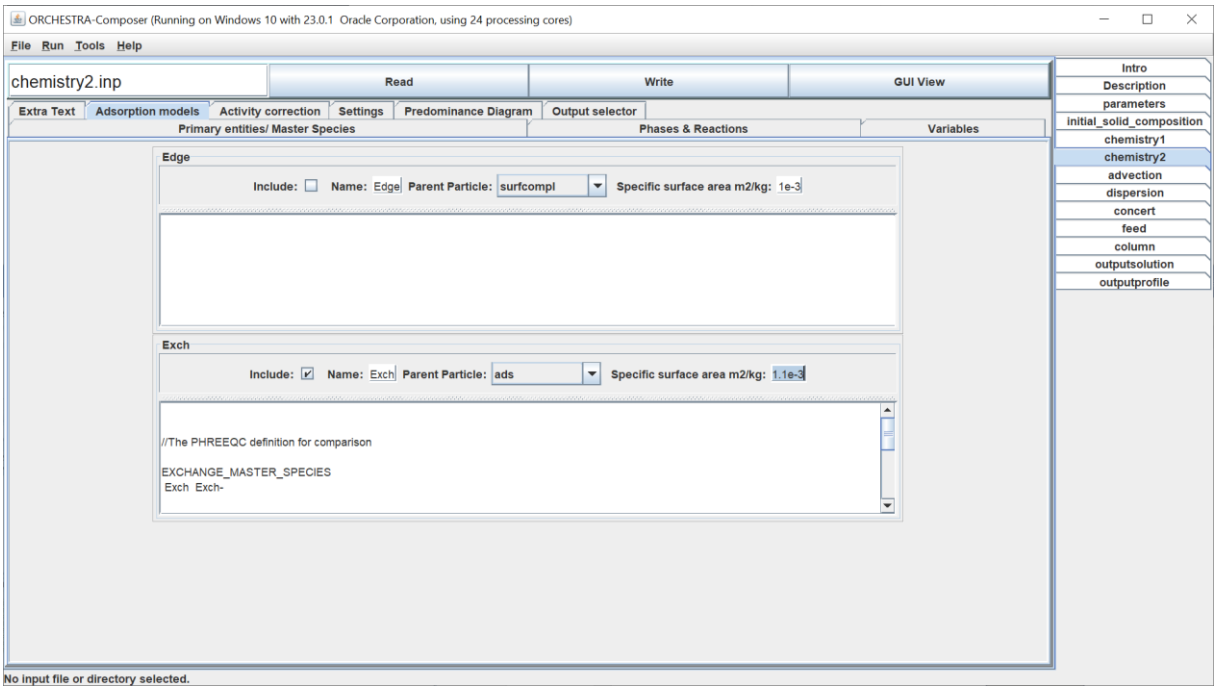




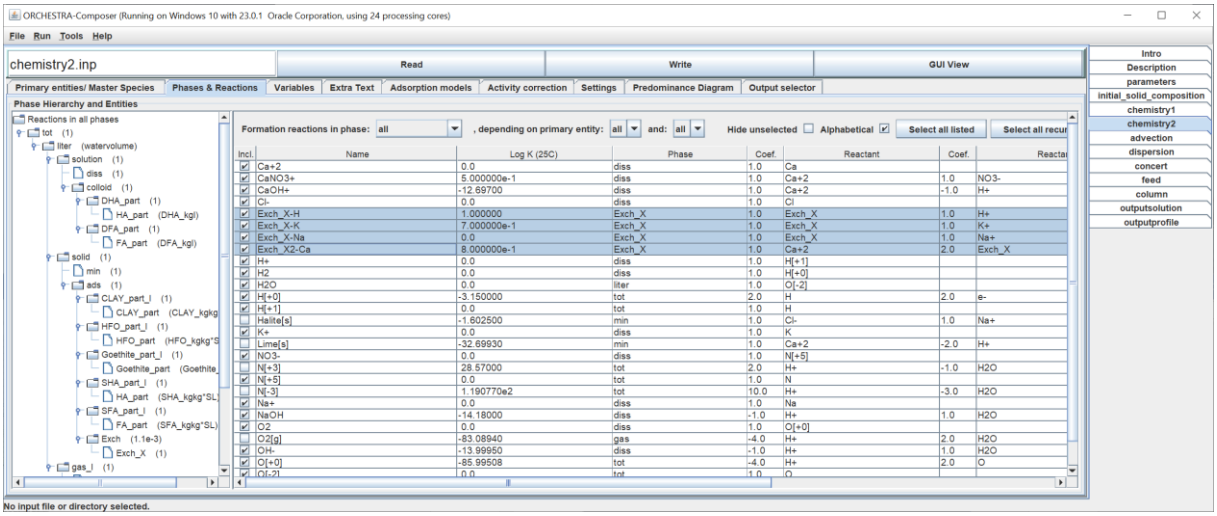
Both solvers contain the same set of chemical reactions (including ion exchange reactions):



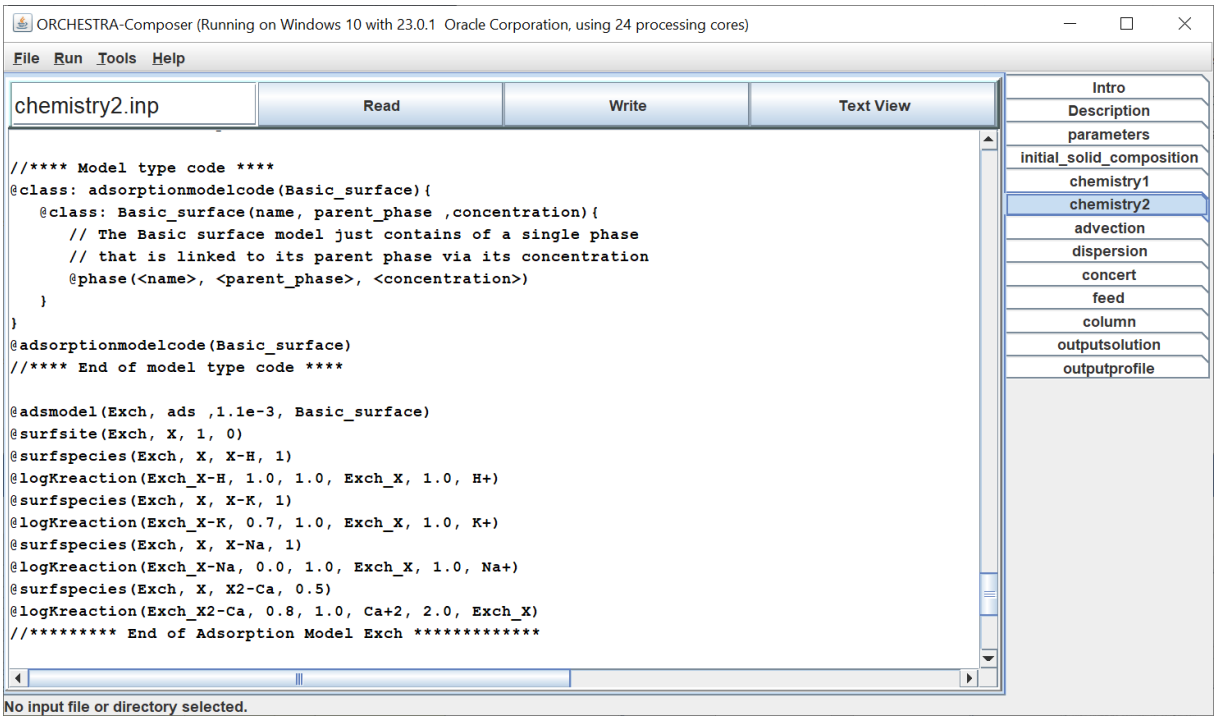
And the same amount of exchange surface (1.1e-3 mol/l)



In contrast with PHREEQC, in the ORCHESTRA input file all (adsorption) reactions are present and copied from the database(s). So the databases are only used when the system is interactively defined, and not required during a run.

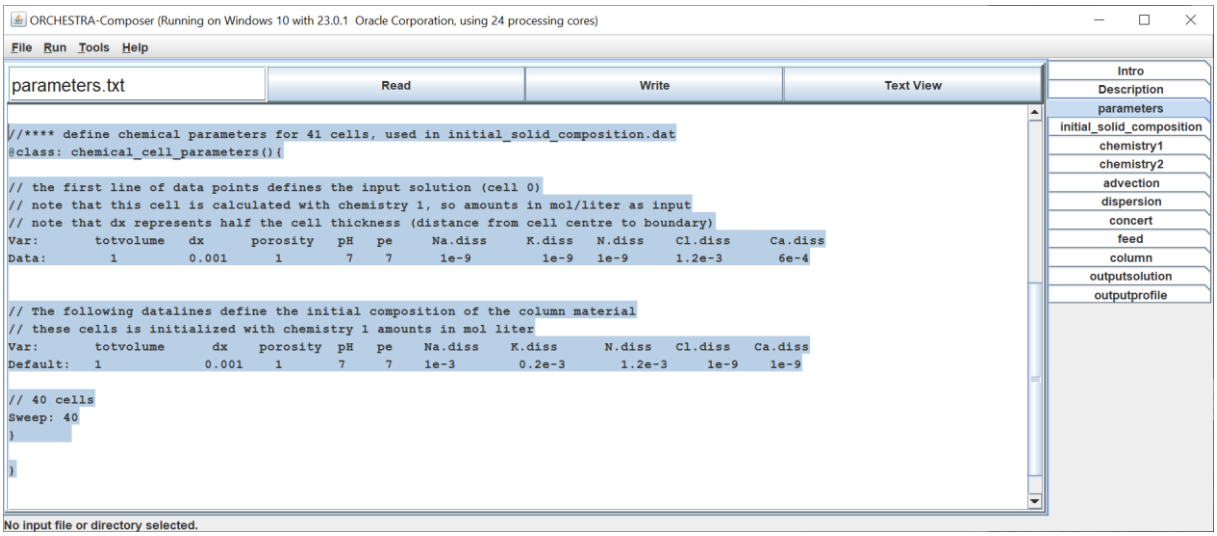


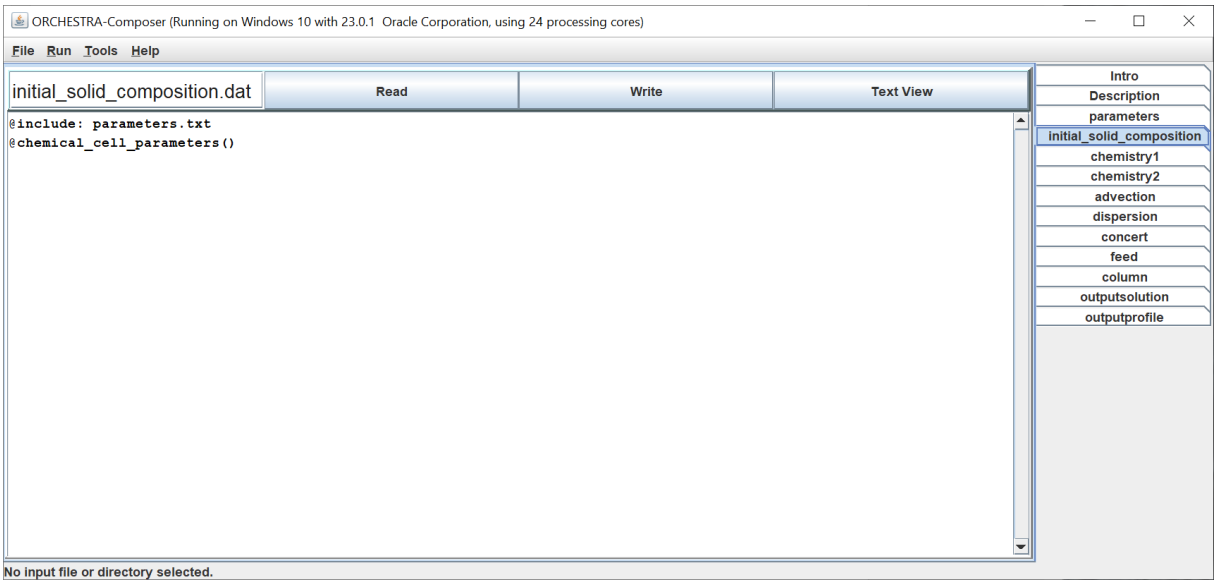
In case of adsorption models, the complete model definition is also copied to the input file. So not only the adsorption reactions, but also the underlying adsorption model code. In case of ion exchange model, this is only a very limited amount of code. Switching to text representation of the GUI View makes this visible.



Definition of chemical input parameters in parameter file

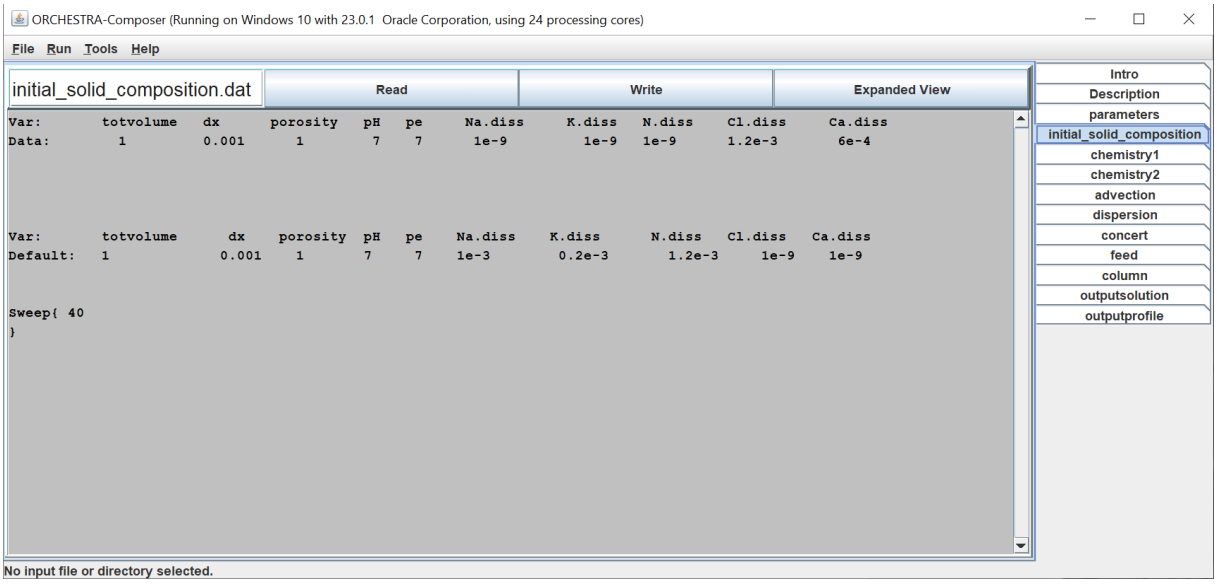
We now have our chemical model defined in the two chemical calculators, and next we can give the values for the initial concentrations in the input file and column. The values are given in the parameter.txt file. Here you find the definition of 41 compositions (for the input solution, cell 0 + 40 column cells) . In this case this definition is stored in a “class” chemical_cell_parameters, that is used in the “initial_solid_composition” input file.



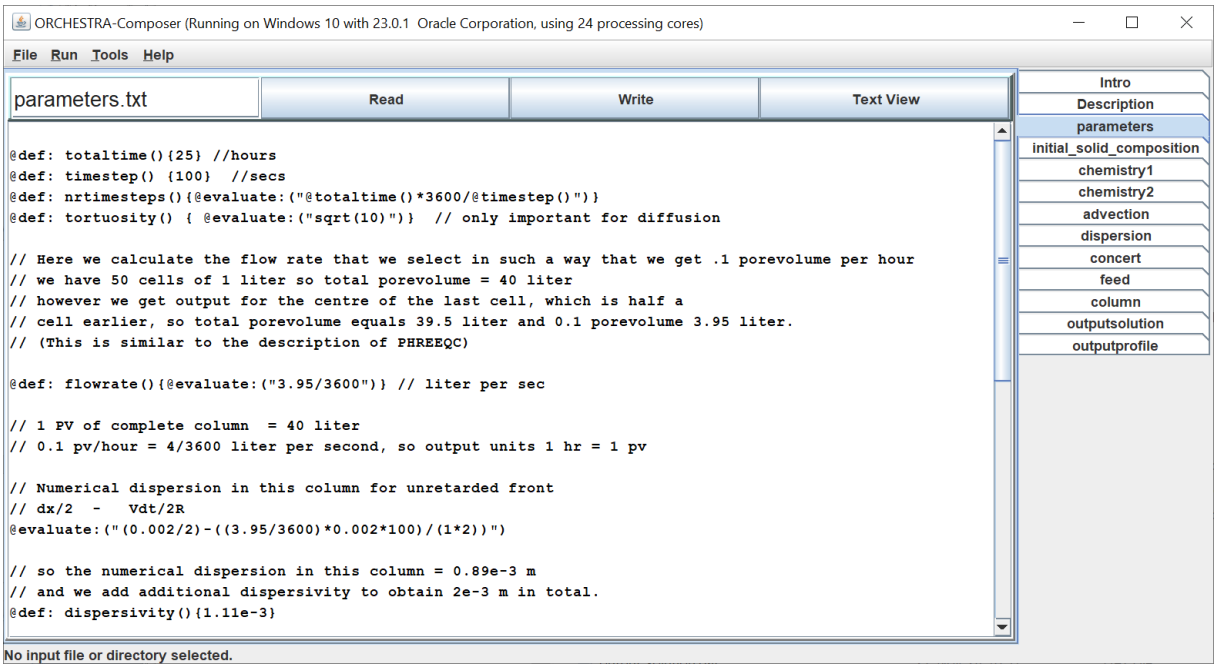


By clicking on text view you can check that the @chemical_cell_parameters() statement is correctly expanded to the full text in the parameters file.

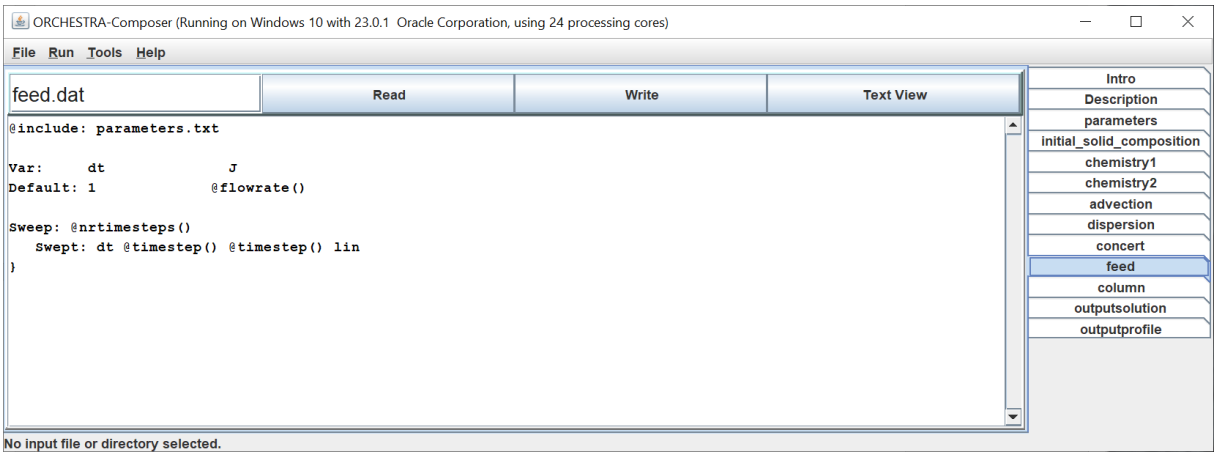
This expanded text is used as input, as all input files are by default expanded.



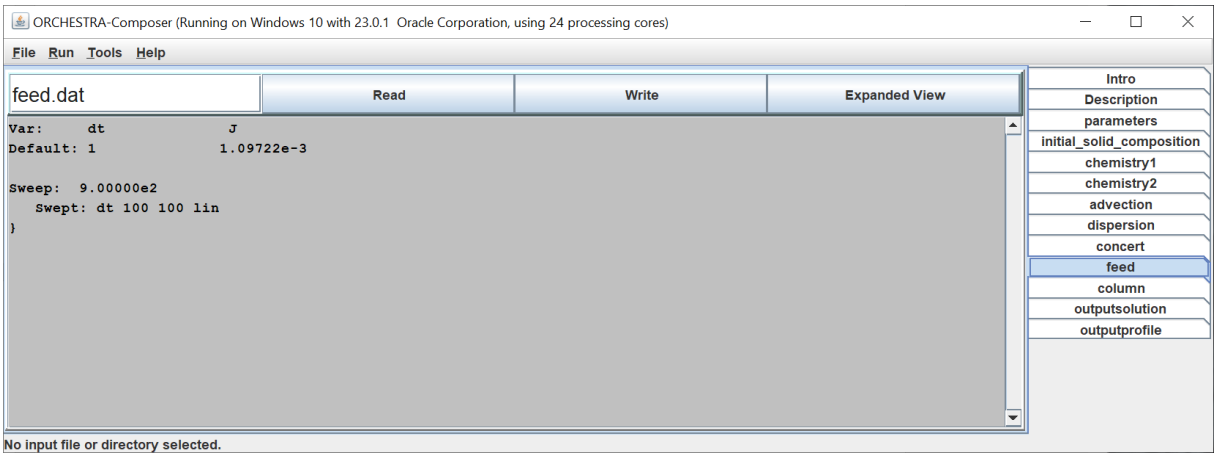
The parameter file also contains several transport parameters



Which are mainly used in the feed.dat file:



Which is also expanded at read in time:



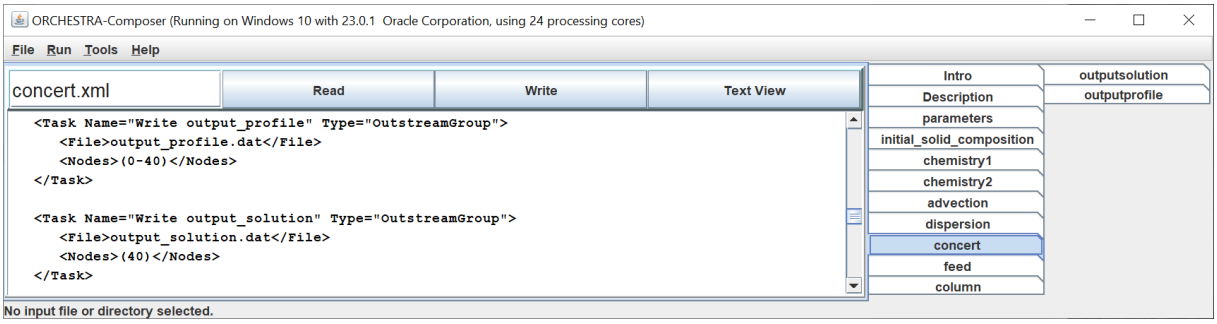
And contains the timestep (100 sec) , total number of timesteps (900) and flow speed (liter / sec).

Definition output variables and frequency in output file

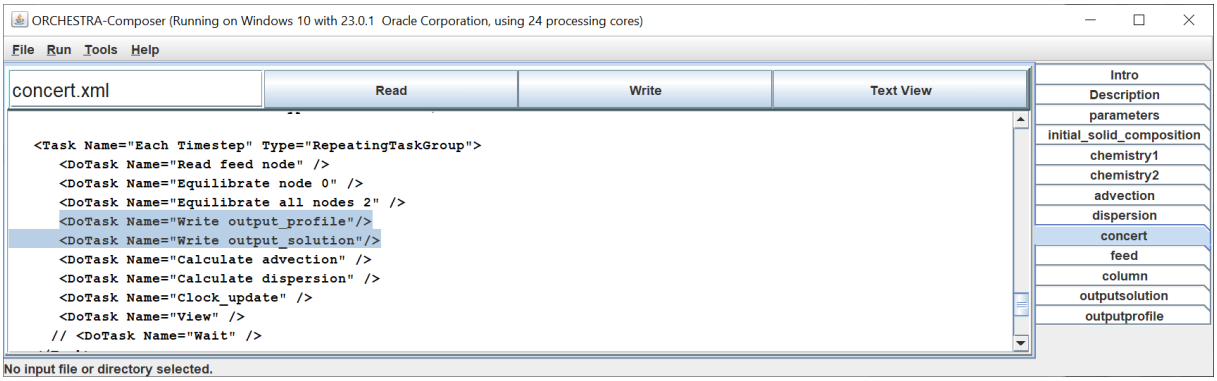
The system has two different output files

1) Outputsolution.dat that contains the composition of the last cell and is called every timestep
2) Outputprofile.dat that contains the composition of all cells

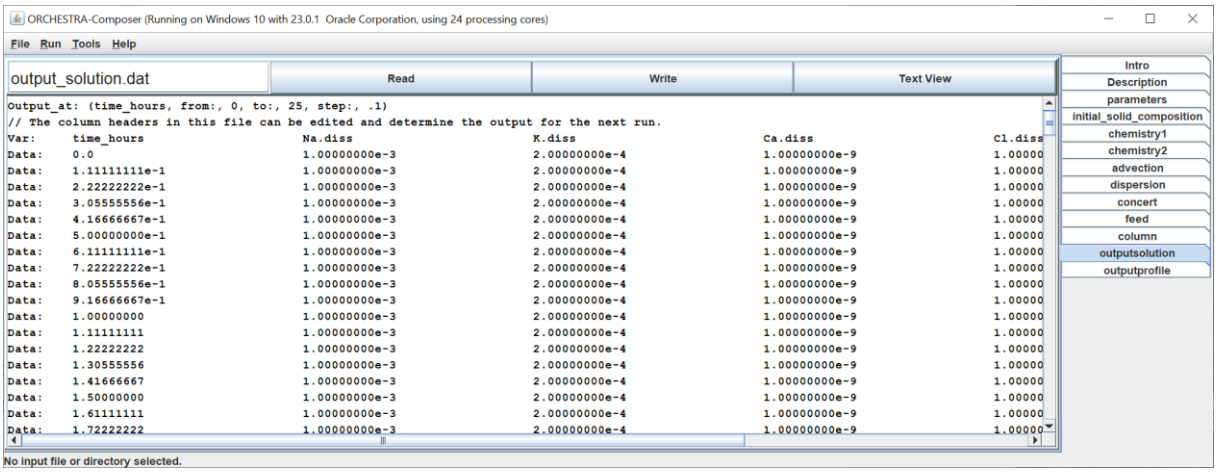
In the concert file these two files are defined as follows:



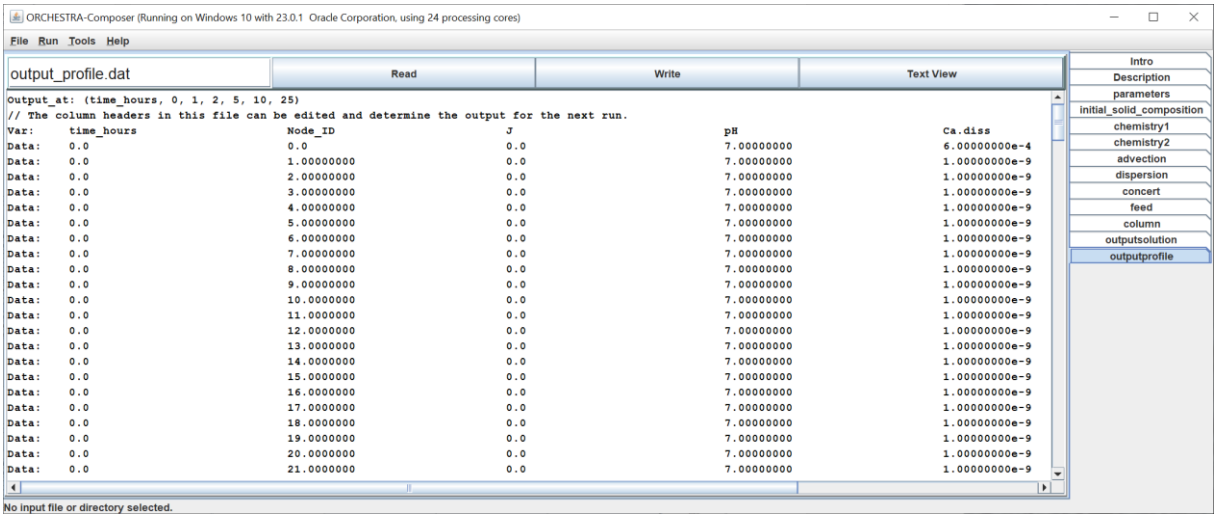
And called every timestep:



The content and frequency of the output is defined within the output file. So for output solution we get output for the listed variables every 0.1 hour from 0 to 25 hours.



For output of the complete profile we specify the times at which we want output:



So please note that output times and output variables are defined **within the output files themselves!!**

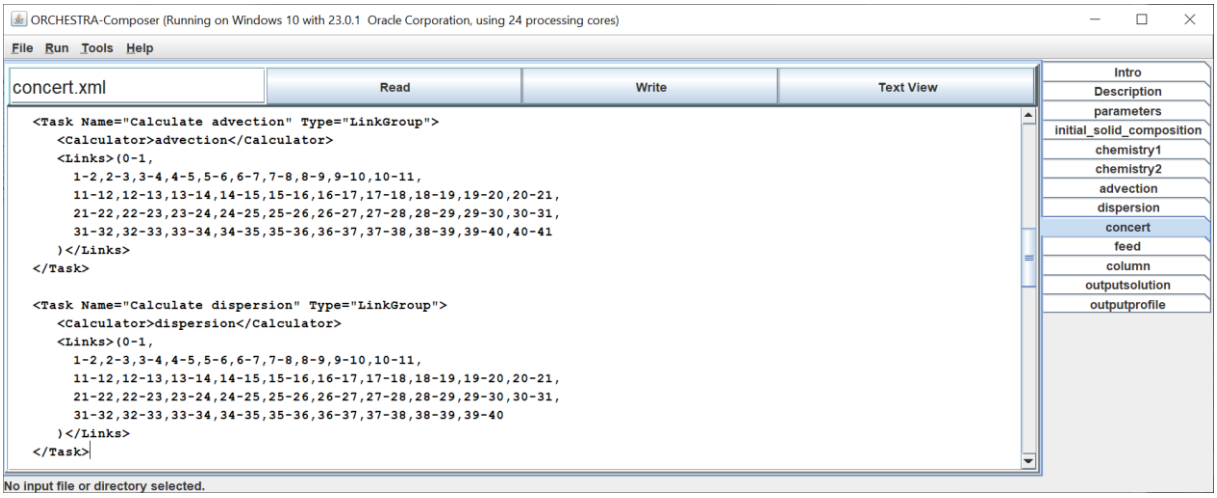
Definition of Advection and Dispersion

In the concert.xml file you will also find two transport tasks defined; advection and dispersion.

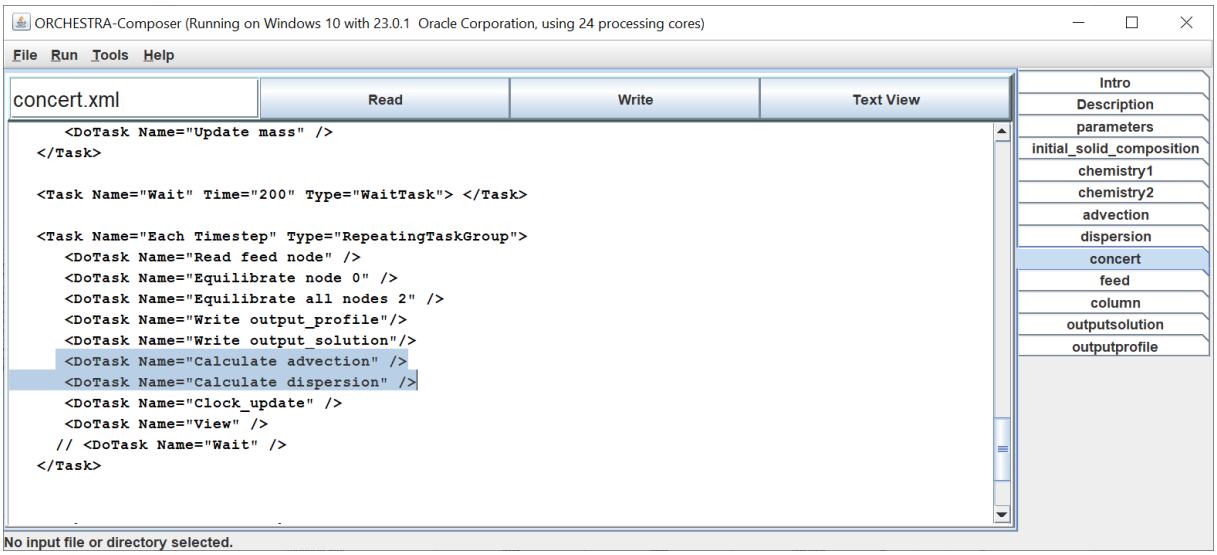
These tasks are defined as a set of connections of links between cells. For each of these connections some calculations are defined in the advection.inp and dispersion.inp text files.

Note that the connections for the advection and dispersion task are not exactly the same: the dispersion task does not have the connection with the final node 41 (waste bucket or sink).

This waste bucket is only used by advection to remove all mass that leaves cell 40. Advection does not look at the content of cell 41, so cell 41 does not contain realistic concentrations (is not calculated or updated). However, dispersion (like diffusion) is a function of the concentrations in the two connected cells, so in this case the (unrealistic) concentrations in cell 41, would result in errors. (you can try this for yourself by adding the link 40-41 and check the results.



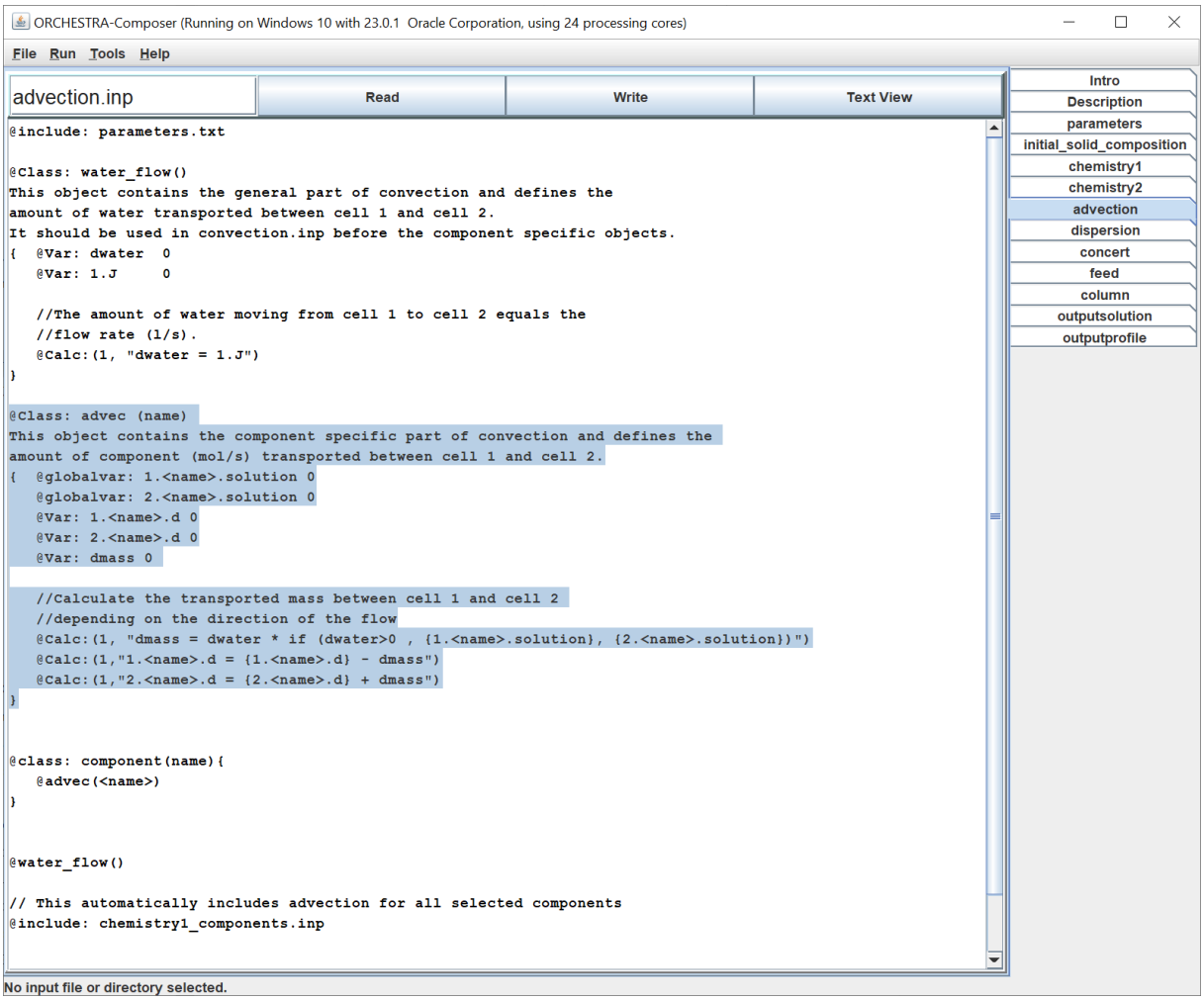
Calculation of advection and dispersion happens every timestep (after all nodes/cells are equilibrated).



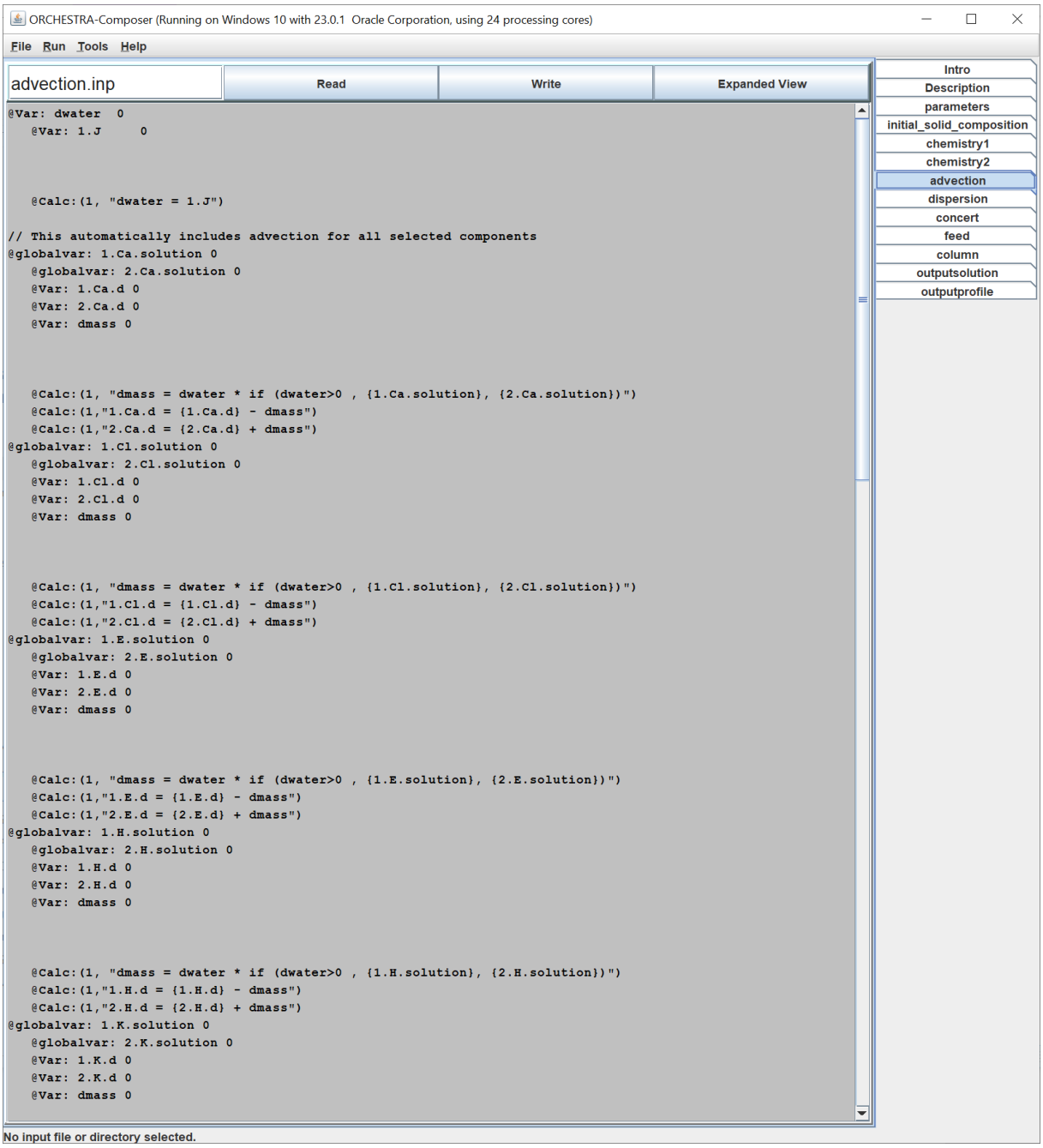
The actual calculations that take place for advection and dispersion are defined in the advection.inp and dispersion.inp input files.

Here a class “advec” is defined that calculates the mass transport (calculates delta masses in both connected cells) using the amount of water and the concentration of the incoming solution as input.

This advec class is called of each primary entity / master species.



You can check this by clicking on the “Text view button”

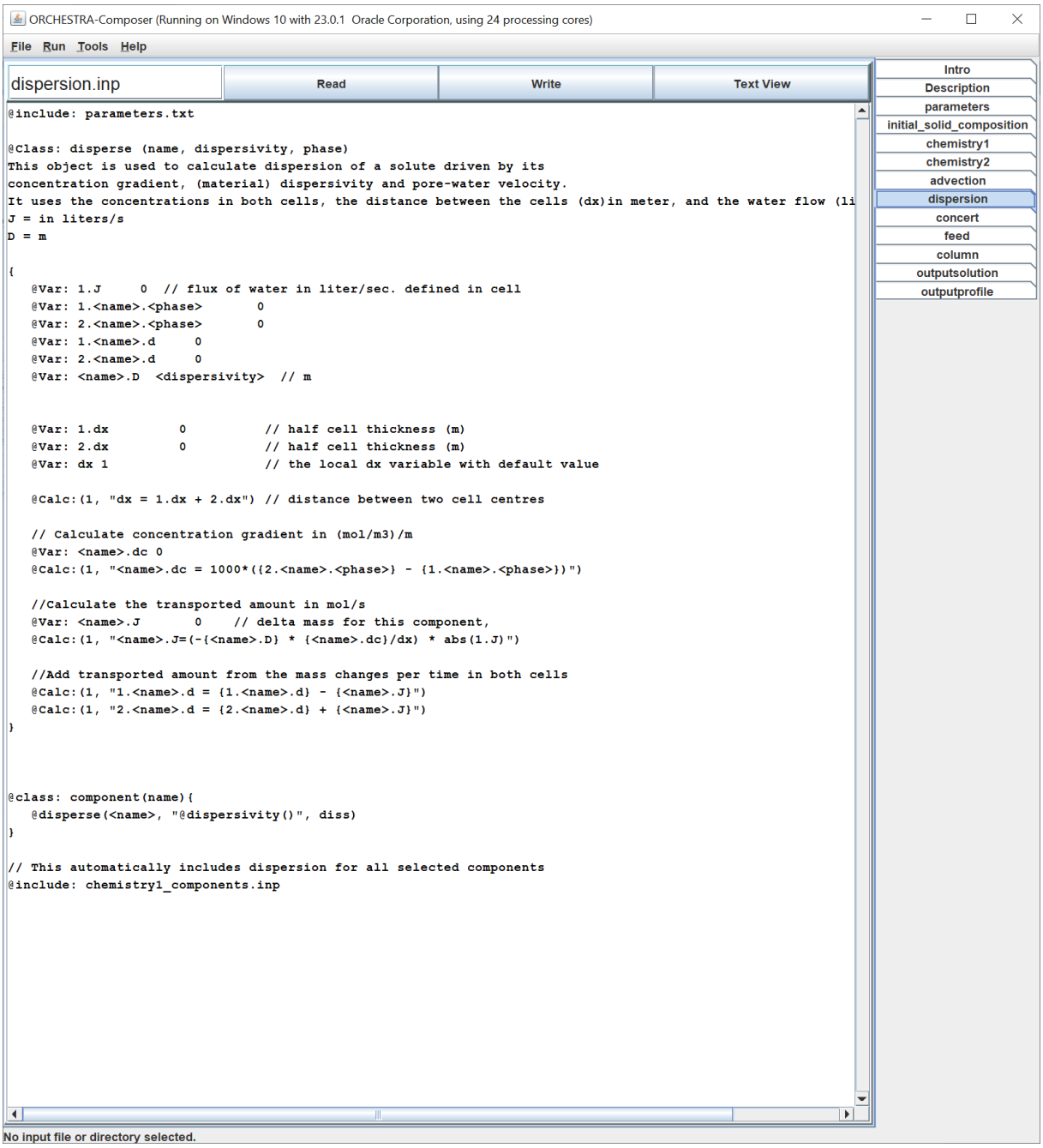


Here you can see the expressions written out for all the primary entities. This demonstrates that in case of ORCHESTRA not only for the chemical part, but also for the transport systems you have access to all the equations/expression involved at the lowest level.

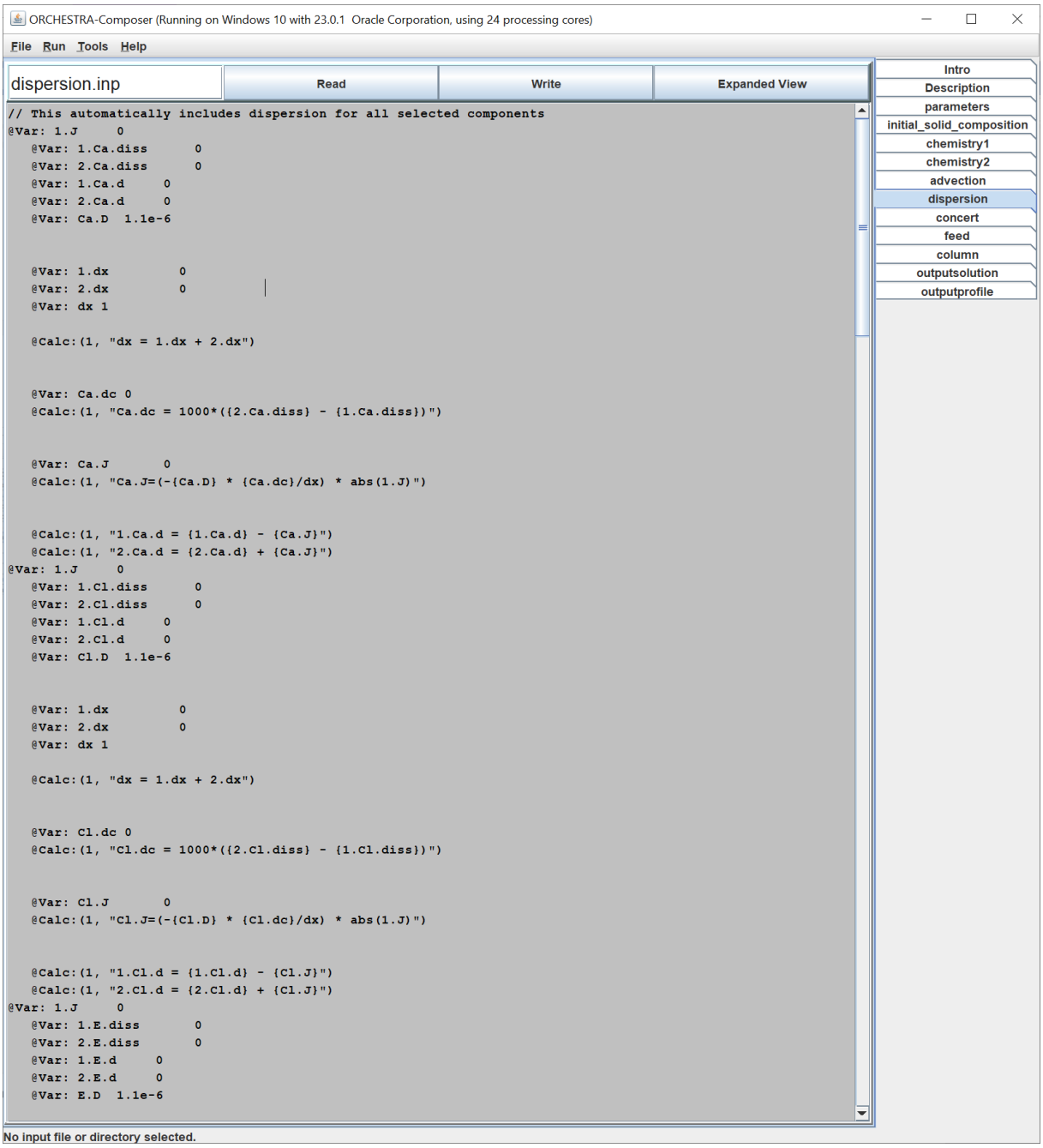
ORCHESTRA uses the same calculator/ solver for chemical as well as transport calculations!

So the syntax and use of objects/templates etc. is the same!

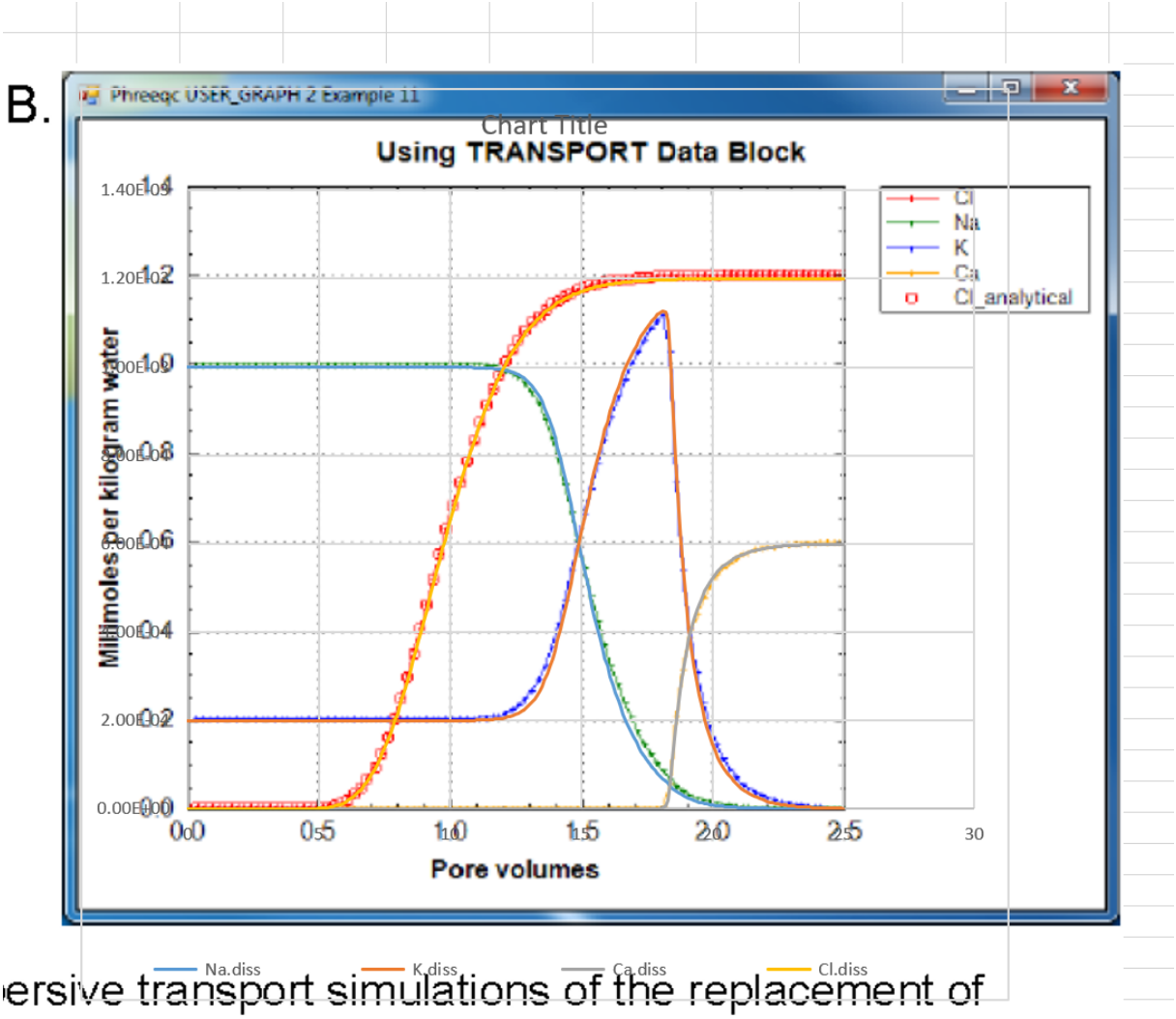
The definition of dispersion is very similar to that of advection:



In this case the amount of mass transfer is determined by concentration difference between the two connected cells, the distance between the cells, and the water flow. Note that as a result dispersion stops when the water flow is stopped (in contrast with molecular diffusion) . Again you can check the expansion process by clicking Text view to see the literal calculations:


ORCHESTRA-Composer (Running on Windows 10 with 23.0.1 Oracle Corporation, using 24 processing cores)
File Run Tools Help
dispersion.inp
Read
Write
Expanded View
Intro
Description
parameters
initial_solid_composition
chemistry1
chemistry2
advection
dispersion
concert
feed
column
outputsolution
outputprofile
// This automatically includes dispersion for all selected components
@Var: 1.J 0
@Var: 1.Ca.diss 0
@Var: 2.Ca.diss 0
@Var: 1.Ca.d 0
@Var: 2.Ca.d 0
@Var: Ca.D 1.1e-6
@Var: 1.dx 0
@Var: 2.dx 0
@Var: dx 1
@Calc:(1, "dx = 1.dx + 2.dx")
@Var: Ca.dc 0
@Calc:(1, "Ca.dc = 1000*({2.Ca.diss} - {1.Ca.diss})")
@Var: Ca.J 0
@Calc:(1, "Ca.J=(-{Ca.D} * {Ca.dc}/dx) * abs(1.J)")
@Calc:(1, "1.Ca.d = {1.Ca.d} - {Ca.J}")
@Calc:(1, "2.Ca.d = {2.Ca.d} + {Ca.J}")
@Var: 1.J 0
@Var: 1.Cl.diss 0
@Var: 2.Cl.diss 0
@Var: 1.Cl.d 0
@Var: 2.Cl.d 0
@Var: Cl.D 1.1e-6
@Var: 1.dx 0
@Var: 2.dx 0
@Var: dx 1
@Calc:(1, "dx = 1.dx + 2.dx")
@Var: Cl.dc 0
@Calc:(1, "Cl.dc = 1000*({2.Cl.diss} - {1.Cl.diss})")
@Var: Cl.J 0
@Calc:(1, "Cl.J=(-{Cl.D} * {Cl.dc}/dx) * abs(1.J)")
@Calc:(1, "1.Cl.d = {1.Cl.d} - {Cl.J}")
@Calc:(1, "2.Cl.d = {2.Cl.d} + {Cl.J}")
@Var: 1.J 0
@Var: 1.E.diss 0
@Var: 2.E.diss 0
@Var: 1.E.d 0
@Var: 2.E.d 0
@Var: E.D 1.1e-6
No input file or directory selected.

In the accompanying Excel spreadsheet the ORCHESTRA output is plotted on top of the PHREEQC graph, (I will make a better graph later) and there is excellent agreement between the PHREEQC and ORCHESTRA results!



The zipped folder contains all the necessary files (PHREEQC input file, ORCHESTRA complete model and input files) to reproduce the calculations yourself.