Appendix

This appendix provides general information for the main classes in the developed interface. Because COMSOL API is designed for Java, the interface was also developed in Java; therefore, the C++ methods of GEMS needed to be translated using the Java Native Interface (JNI). This was achieved mostly though wrappers that could be read from Java libraries. Wrappers were developed for each GEMS3K method.

Figure A1 presents main processes of the developed interface following the time marching algorithm that is presented in Fig. 2. Table A1 provides information about main classes and processes that were developed as part of the interface, as well as GEMS and COMSOL classes and their methods/data structures.

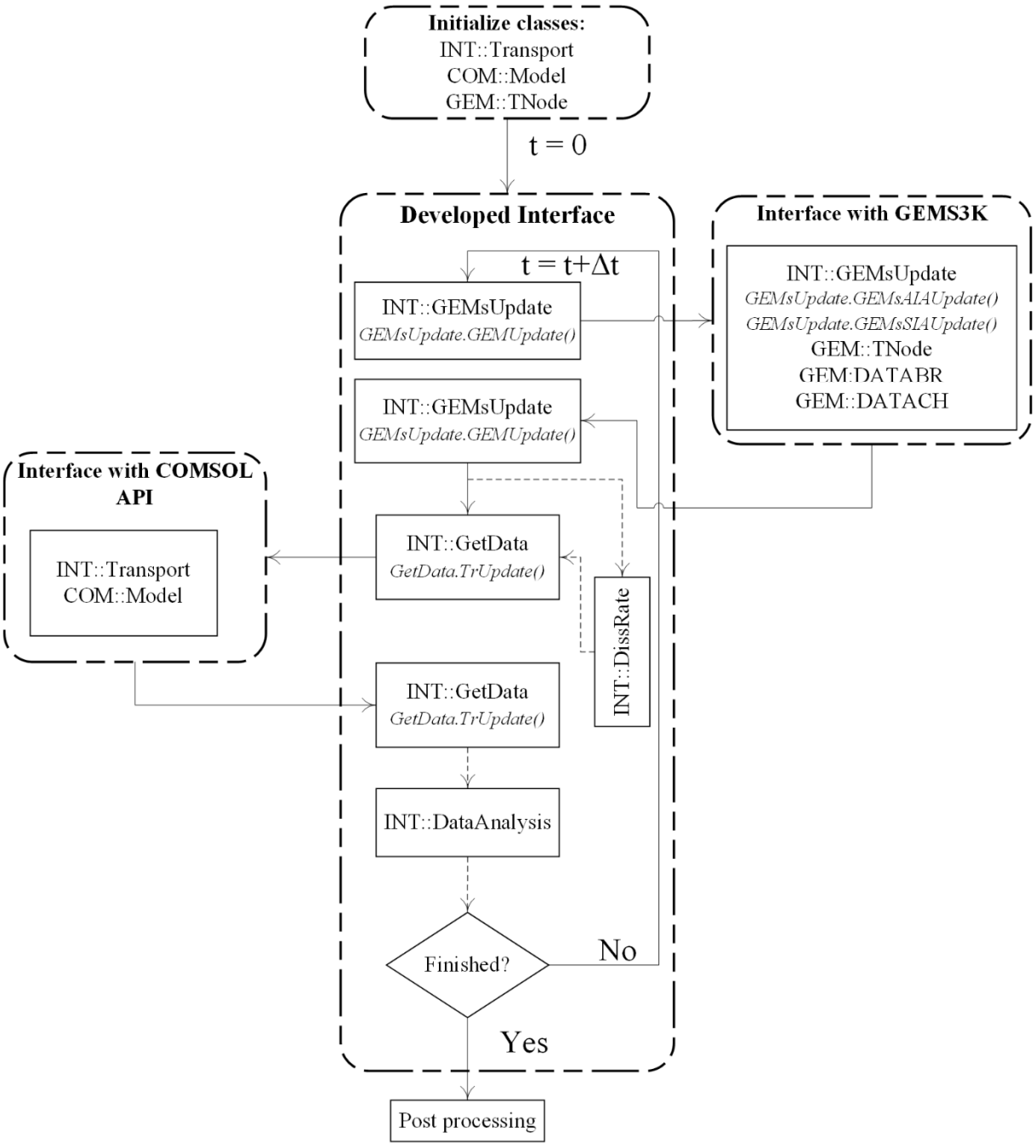

Figure A1: Main classes and methods that are used in the time marching scheme of the developed interface.

Table A1: Description of main classes and methods of the developed interface

Developed Java class(es)/ methods	**Description**	**Used GEMS class(es)/ methods**	**Used COMSOL class(es)/ methods**
Class(es): INT::Transport **Main methods:** Transport.run() Trasnport.comsolOneTimeStepRun() Transport.bcChange() Transport.trUpdate()	The main class that interfaces with COMSOL API. It is used to setup PDEs, geometry of the domain, discretization, material properties, and boundary conditions.	**Class(es):** GEM::TNode GEM:: DATACH GEM:: DATABR **Main methods*:** DATACH.pCSD DATABR.pCNode TNode.cTC() TNode.cTK() TNode.cP() TNode.cMs() TNode.cVs() TNode.setNodeHandle()	**Class(es):** COM::Model **Main methods:** Model.ModelUtil.create() Model.geom() Model.param() Model.variable() Model.mesh() Model.physics() Model.study() Model.sol() Model.solverSequence Model.result().numerical().create() Model.func()
Class(es): INT::Init **Main methods:** Init.initialize()	Initializes the GEMS models for all domains; gets the chemical species, phases and elements; and re-arranges them in the formats that can be introduced as COMSOL state variables.	**Class(es):** GEM::TNode **Main methods:** TNode.IC_name_to_xCH() TNode.DC_name_to_xCH()	

		TNode. Ph_name_to_xCH() TNode.IC_name_to_xDB() TNode.DC_name_to_ xDB() TNode. Ph_name_to_ xDB() TNode.Get_nDC() TNode.Ph_Volume()	
Class(es): INT:: GEMsUpdate **Main methods:** GEMsUpdate.GEMsAIAUpdate() GEMsUpdate.GEMsSIAUpdate() GEMsUpdate.GEMUpdate()	Performs the thermodynamic analyses with the given GEMS input data. The Automatic Initial Approximation (AIA) method is used for the first analysis (GEMsAIAUpdate). For the following analyses a Smart Initial Approximation is used based on the nodal history (GEMsSIAUpdate). The resulting data from GEMS are stored in its data structures. TrUpdate stores these data for post processing purposes as well.	**Class(es):** GEM::TNode GEM:: DATACH GEM:: DATABR **Main methods:** TNode. GEM_restore_MT() TNode.GEM_to_MT() TNode.GEM_run(true/false) TNode.GEM_set_MT() TNode.BAD_GEM_AIA_get() TNode.BAD_GEM_SIA_get() DATACH.pCSD DATABR.pCNode	
Class(es): INT:: GetData **Main methods:**	Updates variables and functions that change in each analysis step. These include diffusion coefficients, porosity, the boundary condition functions,	-	-

GetData.getTemp() GetData.getPressure() GetData.getChemicalSpecies() GetData.getDiffusion() GetData.getPorosity() GetData.getStochimetry() GetData.getIntFunc() GetData.TrUpdate()	additional stoichiometry data, etc.		
Class(es): INT:: DissRate **Main methods:** DissRate.dissolvedMineral() DissRate. getDissRateMineral() DissRate.dissolvedCEM() DissRate.getDissRateCEM() DissRate.set_adll() DissRate.set_adul()	Obtains the dissolved/precipitated minerals based on different kinetics that are defined for the analysis.	-	-
Class(es): INT:: DataAnalysis **Main methods:** DataAnalysis.writeArray() DataAnalysis.exportData()	Writes the output files during the analysis based on the required data and time steps.	-	-