CONCEPTUALIZATION AND SOFTWARE DEVELOPMENT OF A SIMULATION ENVIRONMENT FOR PROBABILISTIC SAFETY ASSESSMENTS OF RADIOACTIVE WASTE REPOSITORIES

Doctoral Thesis

(Dissertation)

To be awarded the degree

Doctor rerum naturalium (Dr. rer. nat.)

submitted by

Javad Ghofrani, M.Sc.

From Iran/ Razan

approved by the Faculty of Energy and Management Science

Clausthal University of Technology,

Date of oral Examination

26 May 2016

Dean

Prof. Dr. rer. pol. Inge Wulf

Supervising tutor

Prof. Dr. rer. nat. Klaus-Jürgen Röhlig

Reviewer

Prof. Dr. rer. nat. Sven Hartmann

This thesis has been written as part of the work of the research platform ENTRIA - Disposal Options for Radioactive Residues: Interdisciplinary Analyses and Development of Evaluation Principles. ENTRIA is a joint research project financed by the German Ministry of Education and Research (BMBF, SUPPORT CODE 02S9082A).

Federal Ministry
of Education
and Research

SUPPORT CODE 02S9082A

DISPOSAL OPTIONS FOR RADIOACTIVE RESIDUES:
INTERDISCIPLINARY ANALYSES AND
DEVELOPMENT OF EVALUATION PRINCIPLES

EIGENSTÄNDIGKEITSERKLÄRUNG

Hiermit versichere ich, dass ich die vorliegende Dissertation selbständig und nur mit den angegebenen Hilfsmitteln verfasst habe. Alle Passagen, die ich wörtlich aus der Literatur oder aus anderen Quellenübernommen habe, habe ich deutlich als Zitat mit Angabe der Quelle kenntlich gemacht.

STATUTORY DECLARATION

I declare that I have authored this thesis independently, that I have not used other than the declared sources / resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Abstract

Uncertainty and sensitivity analysis of complex simulation models are prominent issues, both in scientific research and education. ReSUS (Repository Simulation, Uncertainty propagation and Sensitivity analysis) is an integrated platform to perform such analysis with numerical models that simulate the THMC (Thermal Hydraulical Mechanical and Chemical) coupled processes via different programs, in particular in the context of safety assessments for radioactive waste repositories. This thesis presents the idea behind the software platform ReSUS and its working mechanisms. Apart from the idea and the working mechanisms, the thesis describes applications related to the safety assessment of radioactive waste disposal systems.

In this thesis, previous simulation tools (including the preceding version of ReSUS) are analyzed in order to provide a comprehensive view of the state of the art. In comparison to this state, a more sophisticated software tool is developed here, which provides features which are not offered by previous simulation tools. To achieve this objective, the software platform ReSUS provides a framework for handling probabilistic data uncertainties using deterministic external simulation tools, thus enhancing uncertainty and sensitivity analysis. This platform performs probabilistic simulations of various models, in particular THMC coupled processes, using stand-alone deterministic simulation software tools. The complete software development process of the ReSUS Platform is discussed in this thesis.

ReSUS components are developed as libraries, which are capable of being linked to other code implementations. In addition, ASCII template files are used as means for uncertainty propagation into the input files of deterministic simulation tools. The embedded input sampler and analysis tools allow for sensitivity analysis in several kinds of simulation designs.

The novelty of the ReSUS platform consists in the flexibility to assign external stand-alone software tools (regardless of source code availability) to probabilistic simulations. Furthermore, the ReSUS platform implements a simple scientific workflow management system to handle the data and process flow between several external simulation programs automatically, thus allowing for the creation of chains of simulation models. In addition, the multilayer architecture of ReSUS platform provides a number of different interfaces to connect with other tools in the field of uncertainty and sensitivity analysis.

Zusammenfassung

Unsicherheits- und Sensitivitätsanalysen von komplexen Simulationsmodellen sind sowohl in der Forschung als auch in der Lehre bedeutende Themen. ReSUS (Repository Simulation, Uncertainty propagation and Sensitivity analysis) ist eine integrierte Plattform für solche Analysen unter Nutzung von numerischen Modellen bzw. Programmen zur Simulation von Thermo-Hydraulisch-Mechanisch-Chemisch (THMC) gekoppelten Prozessen, insbesondere in Zusammenhang mit Sichrheitsanalysen für Endlager radioaktiver Abfälle.

Die vorliegende Arbeit stellt die Idee hinter ReSUS sowie die Arbeitsmechanismen vor. Zudem erfolgt die Beschreibung einer beispielhaften Anwendung mit Bezug zur Sicherheitsbewertung von Endlagern für radioaktive Abfälle.

In der vorliegenden Arbeit wird darüber hinaus ein umfassender Überblick über weitere gängige einschlägige Simulation- und Analysestools (einschließlich der Vorgängerversion von ReSUS) und damit über den Stand von Wissenschaft und Technik gegeben. Die Entwicklung des Software-Tools ReSUS orientiert sich an diesem Stand und soll, im Vergleich zu diesen Programmen, ergänzende Funktionalität zur Verfügung stellen. Diese Software-Plattform ermöglicht dahingehend die probabilistische Behandlung von Datenunsicherheiten durch die Nutzung deterministischer Fremdcodes, und bietet damit verbesserte Möglichkeiten zur Unsicherheits- und Sensitivitätsanalyse. Mit der Plattform können durch die Nutzung eigenständiger deterministischer Software-Tools unterschiedliche Modelle probabilistisch simuliert werden, insbesondere THMC-gekoppelten Prozesse. Der gesamte Software-Entwicklungsprozess der ReSUS-Plattform wird in der vorliegenden Arbeit diskutiert.

Die ReSUS-Komponenten wurden als Bibliotheken entwickelt, die eine Verbindung mit anderen Code-Implementierungen ermöglichen. Darüber hinaus werden ASCII-Datei-Templates verwendet, um die Informationen über die Unsicherheiten zwischen den jeweiligen einzelnen Simulationen bzw. Fremdcodes zu übertragen. Es können Ketten von Simulationsmodellen angelegt und ausgeführt werden. Die eingebetteten Stichprobenerzeuger und Analyse-Tools ermöglichen Sensitivitätsanalysen auf verschiedene Arten von Simulation-Designs.

Das Novum der ReSUS Plattform besteht zusammenfassend in deren Flexibilität, ursprünglich geschlossen arbeitende Stand-alone-Software-Werkzeuge (auch unabhängig von der Quellcode-Verfügbarkeit) in globale (probabilistische) Simulationen zu integrieren. Darüber hinaus bietet die ReSUS Plattform ein einfaches System, welches ein automatisches, wissenschaftliches Workflow-Management des Daten- und Prozessablaufs zwischen den zugewiesen Simulationsprogrammen ermöglicht. Zusätzlich bietet die Mehrschichtarchitektur der ReSUS Plattform eine Reihe von verschiedenen Schnittstellen zu anderen Werkzeugen im Bereich der Unsicherheits- und Sensitivitätsanalyse.

Acknowledgements

I would like to thank my supervisor Prof. Dr. Klaus-Jürgen Röhlig for his support and encouragement during this project. At many stages in the course of this research project I benefited from his advice, particularly so when exploring new ideas. His careful editing contributed enormously to the production of this thesis.

A very special thanks goes out to my co-supervisor Prof. Dr. Sven Hartmann, who has been always played a key role in encouraging and coordinating this whole project and keeping me abreast of the scientific developments and his support during the course of this study.

I would also like to thank Dr. Elmar Plischke, Dr. Xiaoshuo Li and Willy Ciecior for their scientific help during the research program.

Furthermore, I would like to thank all members of the ENTRIA project for their help and support of my PhD work. I also apologize to everyone who may not be mentioned personally here.

Finally, I would like to thank Mostafa Lotfalipour, Arezu Bozorgmehr and Mohammad Divband Soorati for taking time out from their busy schedule to serve as my external reader and editing assistance.

In conclusion, I recognize that this research would not have been possible without the financial assistance of ENTRIA (BMBF, SUPPORT CODE 02S9082A) and the Institute of Disposal Research at the Clausthal University of Technology, and express my gratitude to the project sponsors.

Table of Contents

List of tables

List of Figures

1 Introduction

One of the purposes of modelling and simulation in the safety assessments for radioactive waste disposal systems (Ruark, Niemann, Greimann, & Arabi, 2011) is to analyze the thermally, hydraulically, mechanically and chemically (THMC) coupled processes in different components of a radioactive waste repository using so-called process models. In contrast, integrated or system-level models are used to assess the performance of the disposal system as a whole and to evaluate the performance and potential impact by safety indicators such as annual effective dose or risk over the whole assessment time frame.

In the presence of uncertainty, results also become uncertain. We concentrate our efforts on data uncertainty by embedding the deterministic numerical simulations in a probabilistic framework. The progress of sample generation, simulation and analyzing results is an established method to study the impact of input parameter uncertainty. The software tools, which implement this method for safety assessments of radioactive waste-disposal systems, are mostly designed for a certain model or a problem and cannot be used in universally (e.g. for research and educational purposes). These software tools are just equipped with predefined solvers and calculation components so that adding new abilities and functionalities to them is not possible. The preceding version of ReSUS (Li, 2015) is developed to overcome these problems. That version represents an approach to assign third party executables in a probabilistic simulation tool. However, the flexibility and extendibility in that version are limited.

Our software platform, the development of which is presented in this thesis, is as an aid tool for educational and research objectives. It can be utilized for an optimal use of legacy and new third party simulation programs in probabilistic analysis. Additionally, the sensitivity analysis methods provided by the ReSUS platform can be applied to perform safety assessment in a variety of research and educational fields. The ReSUS Platform is an integrated software tool with the ability to generate random samples, integrate stand-alone legacy or new deterministic simulation tools, design workflows for simple and complicated simulation models, perform probabilistic simulations and accomplish uncertainty and sensitivity analysis. This platform makes major changes to the infrastructure of the preceding version of ReSUS (Li, 2015). In the current version, the restrictions concerning assigning new software components are fixed and a GUI-based approach with the ability to build and manage scientific workflows is implemented which provides a flexible way of building probabilistic simulations.

The design of the ReSUS Platform is based on a multilayered architecture. It includes all the pre- and post-processing as well as probabilistic simulation steps. This platform consists of three main components: Core, GUI and Analyzer Tool. The GUI provides the pre-processing facilities which allow the user to create and manipulate the probabilistic simulations. Furthermore, GUI simplifies the design of simulation models and provides a simple access to the Core and Analyzer Tool. The Core component manages and executes the probabilistic simulations. The result of the simulations will be visualized and analyzed by Analyzer Tool. These components use XML files in order to interact with each other.

This work consists of five main parts. The first part includes a brief summary of required knowledge and a survey on the state of the art. The second part discusses the motivation and objective of developing the software platform ReSUS. This platform and its components are introduced from the perspective of the user and the developer in the third part. The fourth part contains some working examples for benchmarking ReSUS. Finally, the fifth part concludes the work done in this thesis.

2 Basic knowledge

This section provides some knowledge background, which is required to understand the basics of the ReSUS Platform.

2.1 Probabilistic simulation

Probabilistic simulations are based on propagating the uncertainty of some probabilistic input parameters of a model and repeatedly performing the simulations. The probabilistic parameters as input of a model are generated randomly according to a probability distribution. The input and output values of a probabilistic simulations can be used for uncertainty and sensitivity analysis.

2.2 Uncertainty Analysis and Sensitivity Analysis

Uncertainty analysis and sensitivity analysis differ in the following sense. The goal of uncertainty analysis is to describe the entire set of possible outputs of a probabilistic simulation, according to their associated probability of occurrence. The sensitivity analysis observes and discovers the effect of changes in model input values on model output values of a probabilistic simulation. The outcome of the uncertainty and sensitivity analysis are used widely for performing safety assessment of a system which is modeled and simulated.

2.3 Coupled Processes

Coupled thermal, hydraulic, mechanical and chemical (THMC) processes consist of more than one T, H, M, or C component, with these components affecting each other in a bidirectional way. There is two types of software packages, which simulate such processes explicitly or implicitly.

On one hand, in the explicit form, each process component is represented as a separate computational component, which can be coupled with other components within the framework using the programming APIs (Application Programming Interfaces). Using such simulation programs requires sophisticated programming and numerical knowledge. The collaboration of these computational components simulates the entire coupled process (for example ALLIANCES (Deville, Montarnal, Loth, & Chavant, 2009)).

In other hand, there are some simulation tools which implement the coupled processes implicitly via numerical calculations methods. This group of simulation programs offer less flexibility by coupling the processes and are also developed to simulate special problems (Level E (Nuclear Energy Agency, 1989)or (Kolditz, Kaiser, Habbar, Rother, & Thorenz, 2003)).

2.4 Scientific Workflow management systems

The term *scientific workflow* describes a sequential progress of work activities and calculations which take place when solving scientific problems. A workflow management system manages the definition, creation, and execution of workflows. In simple form, linear workflows present a sequence of tasks which should be performed in a specified linear order. The first task provides a data object and sends to the next data processing task. This pattern can be used e.g. to model the transfer of radioactive contaminants through different layers of the repository system.

2.5 Overview

The ReSUS Platform implements a linear workflow management system to design and manage the probabilistic simulations. It is capable of assigning deterministic third party simulator programs to the

probabilistic simulations workflows. Both aforementioned methods in 2.3 to simulate the coupled processes can be assigned into the ReSUS Platform as external calculation components which can be also considered as building blocks of a probabilistic simulation workflow. In addition, ReSUS provides post-processing methods to perform uncertainty and sensitivity analysis on the output and input of the probabilistic simulations. The ReSUS platform does not couple any processes biderectionally and only manages the probabilistic simulations with the assigned simulation tools, or chains thereof.

3 State of the Art

We present a survey on the emerging methods and tools to simulate coupled processes and perform uncertainty and sensitivity analysis. We categorized them with regard to their features and working strategies into three groups. The first category includes the software packages, which are intended to perform uncertainty and sensitivity analysis. They are mainly equipped with sampling and statistical tools. The software tools in the second category utilize a workflow management concept to simulate the coupled processes. The third category tools provide a common data exchange framework to support the different simulations. In this category, the simulations will be carried out either by an external code or an internal computational unit.

3.1 JRC SimLab

SimLab is a professional tool for uncertainty and sensitivity analysis (Joint Research Center, 2016). The legacy version of this tool (Tarantola, 2005) is offered as a desktop application with a simple graphical user interface (GUI). Although the version 3.0 of this platform is offered as a development framework that could be usable in C/C++, FORTRAN or MatLab via its provided Application Programming Interfaces (APIs), because of some compatibility problems and development bugs it has been removed from JRC website.

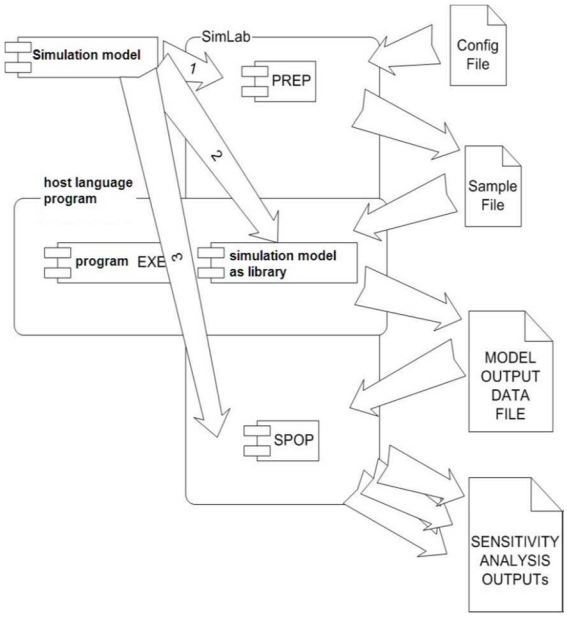


Figure 1. JRC SimLab working methodology (Joint Research Center, 2016)

The simulation steps with JRC SimLab are:

1. Statistical preprocessor (Sample Generation): SimLab provides different sampling methods such as Random sampling, quasi-random LpTau, Replicated Latin Hypercube, Latin Hypercube, classic and extended FAST (Fourier Amplitude Sensitivity Test), Morris and fixed sampled (Giglioli & Saltelli, 2000) to generate samples and save them for simulation input.
2. Model Execution: SimLab is a development framework, which can be used inside the development environments such as MatLab, FORTRAN or C/C++. The executing model should be first implemented in the language of the host environment which is equipped with the provided Application Programming Interfaces (APIs) of SimLab. SimLab also prepares the API needed to manage the input and output of model throughout the run-time. The desktop version allows to assign a simple executable, an MS-Excel spreadsheet or define an mathematical formula as internal model (Joint Research Center, 2016)

3. Statistical postprocessor (Sensitivity analysis (SA) and Uncertainty Analysis (UA)): SimLab provides several sensitivity analysis and uncertainty analysis methods dependent on the input sampling method used. SRC (Standardized Regression Coefficient), PCC (Partial Correlation Coefficient), PEAR (Pearson product moment correlation coefficient), Standardized Rank Regression Coefficients (SRRC's), Partial Rank Correlation Coefficients (PRCC's) and SPEA (Spearman coefficient)) are available if the Random and Monte Carlo sampling method are used. First order and total order sensitivity indices are available for the cases for which FAST and Sobol' are selected for input sampling (Giglioli & Saltelli, 2000).

3.2 DAKOTA Framework

DAKOTA (Adams, et al., 2014) is an acronym for Design Analysis Kit for Optimization and Terascale Applications. Developed first in 1994 by Sandia National Laboratories, it uses iterative systems for executing the analysis codes in order to provide optimization (Giunta A. A., 2002), parameter estimation, uncertainty quantification and sensitivity analysis (Giunta A. A., 2002).

The Dakota toolkit is equipped with algorithms for uncertainty quantification with sampling (Monte Carlo (MC) and Latin hypercube sampling (LHS) (Adams, et al., 2014)), reliability, stochastic expansion, and epistemic methods. It also supports sensitivity analysis with design of experiments and parameter study methods.

DAKOTA is widely used for coupling simulation tools (Larour, et al., 2012) with optimization methods (Hart & Eldred, 1998) (Eldred, Bohnhoff, & Hart, 1999), particularly optimization under uncertainty (Michael S Eldred A. A., 2002) (Giunta A. A., Eldred, Trucano, J., & Steven, 2002) (Giunta A. A., Eldred, Swiler, Trucano, & Wotjkiewicz, 2004) and multilevel parallelism (Giunta A. A., 2002). Basic components of DAKOTA are defined in C++ classes. The Strategy class prepares a control layer on Iteration and Model classes. Model objects are related to simulation codes via the Interface class. The Model class provides input for Simulation codes and gathers their responses. The Interface class provides synchronized and non-synchronized call of executable simulators (Eldred, Bohnhoff, & Hart, 1999). An overview to the structure of this framework has been drawn in Figure 2. Scheduling and synchronizing are implemented in an Application Interface class.

Dakota is capable of uncertainty quantification and methods using LHS (Latin Hypercube sampling), importance sampling and metamodeling approaches.

The DAKOTA Platform is based on a generic interface approach. This approach implements a black-box concept to integrate the external simulation code with the DAKOTA framework. This method can isolate the simulator code from DAKOTA and easily transfers the data between them via input and output text files. (Giunta A. A., 2002). The mechanism of this approach is shown in Figure 2. Running DAKOTA projects consist of four steps (Eldred, Bohnhoff, & Hart, 1999):

1. Defining a Simulation Interface for DAKOTA to work with a simulation code
2. Preparing an input file, this contains environment, model, interface, variables, responses and method specification blocks
3. Initialization and running DAKOTA with the input files prepared
4. Post processing work on DAKOTA results file

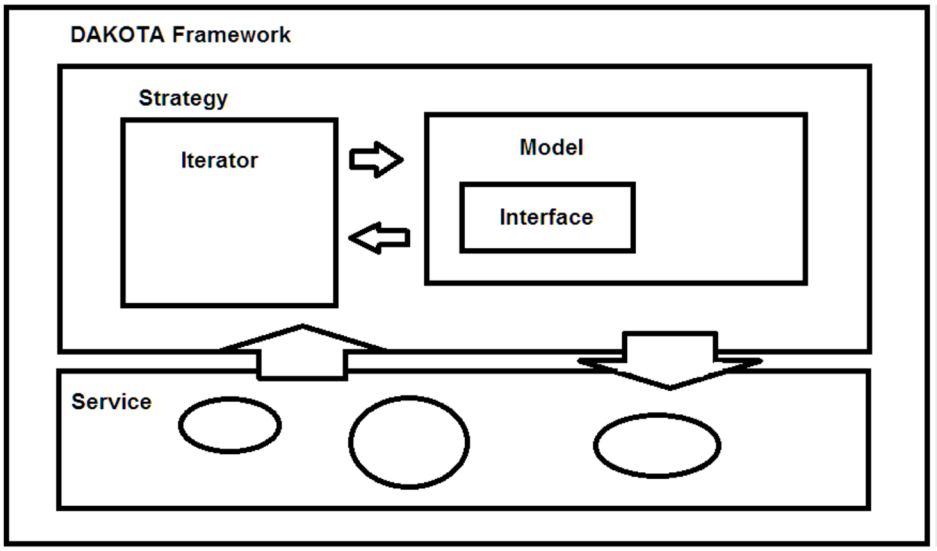


Figure 2. DAKOTA Framework Structure from a developer's point of view. After (Adams, et al., 2015)

DAKOTA supports the parallelization via Message Passing Interface (MPI) (Eldred, Bohnhoff, & Hart, 1999)for working with models, which consist of more than one calculating part. The other important features of this platform are working with template files to generate input files, and providing different sampling methods to generate input values. This framework does not support building a complex chain of models, though.

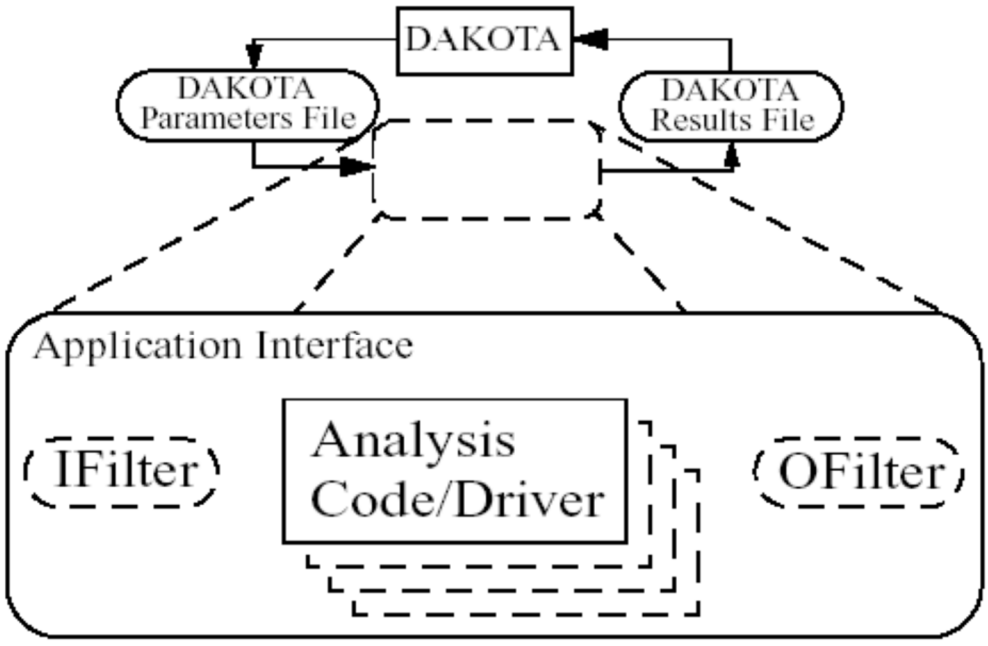


Figure 3. DAKOTA coupling Interface (Sandia Corporation , 2016)

3.3 OpenTURNS

OpenTURNS (open source Treatment of Uncertainty, Risk 'N Statistics) (Andrianov, et al., 2007) has initially been developed as a multiple-access Unix/Linux open source software for uncertainty propagation by probabilistic methods (Baudin, Dutfoy, Iooss, & Popelin, 2015) by an R&D partnership between Electricité de France (EDF), European Aeronautic Defence and Space Company (EADS) and PhiMECA. OpenTURNS as multiple-access software is available as a C++ library, a Python module and

a stand-alone application which is compatible with Windows and Linux (Airbus-EDF-IMACS-Phimeca, 2016).

Uncertainty studies can be performed in OpenTURNS in four main steps with the help of a textual user interface. These steps consist of modeling of uncertainties, of correlation of inputs, propagation of uncertainties and the sensitivity analysis.

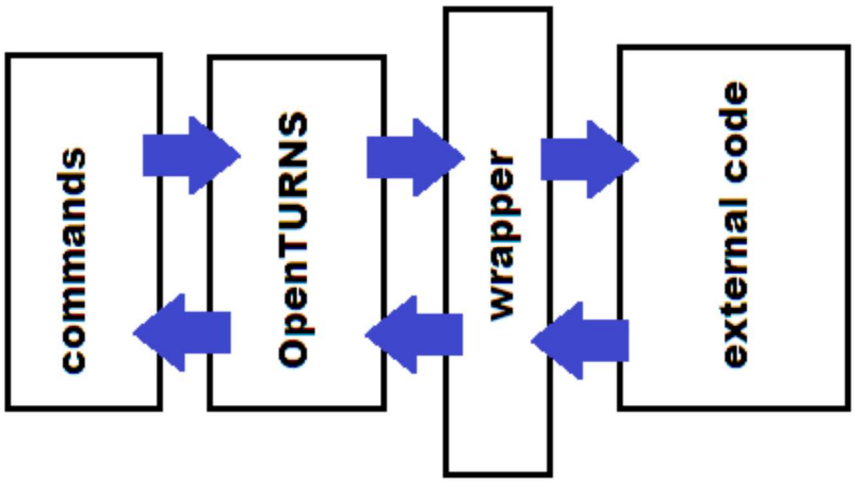


Figure 4. The wrapper in OpenTURNS. after (Airbus-EDF-IMACS-Phimeca, 2016)

This platform is equipped with a collection of mathematical tools in four categories (Baudin, Dutfoy, Iooss, & Popelin, 2015):

1- uncertainty quantification: Modelling of a random vector, Stochastic processes, Statistics estimation, Conditioned distributions, Bayesian Calibration)
2- uncertainty propagation: Min-Max approach, Central tendency, Failure probability estimation, FORM, Monte Carlo, Importance Sampling, Directional Sampling, Subset Sampling
3- sensitivity analysis: Graphical tools, Sampling-based methods, Approximation methods
4- metamodeling methods: Polynomial chaos expansion and the Kriging (Gaussian process regression) approximation

This platform has already been used for a large number of studies in different divisions of EADS such as Airbus or Astrium, using complex software tools as physical model e.g. Abaqus, code-aster and ASERIS (Dutfoy, et al., 2009)The OpenTURNS platform is available as a library in different programming languages. It supports parallel execution of the external codes using a generic wrapper treatment (see Figure 4) (Airbus-EDF-IMACS-Phimeca, 2016). OpenTURNS is also compatible with the SALOME platform (Baudin, Dutfoy, Iooss, & Popelin, 2015).

3.4 Kepler scientific workflow system

Kepler (Altintas, et al., 2004)has been developed based on the Ptolemy 2 framework to provide an integrated framework for realization of design, execution and deployment of scientific workflows (The kepler project, 2015) in order to perform the data analysis. In this framework, the workflows are captured as a set of connected actors. Each actor can execute some computational task such as numerical calculations or data retrieval repetitively. The data flow between actors occurs via their input/output ports and directed connections, which are managed automatically via so-called directors in Kepler. The designed graph of actors and connections is exactly the implementation of scientific workflows. This framework includes a library of actors for different purposes that can be extended by user-defined applications for extra computational tasks. Using actors makes it possible to build a

scientific workflow (The kepler project, 2015) that is enabled to use an external command line application, a remote service or method, or any environments (like MatLab, R or Perl) to interpret or run a script using various data formats (Ludäscher, et al., 2006).

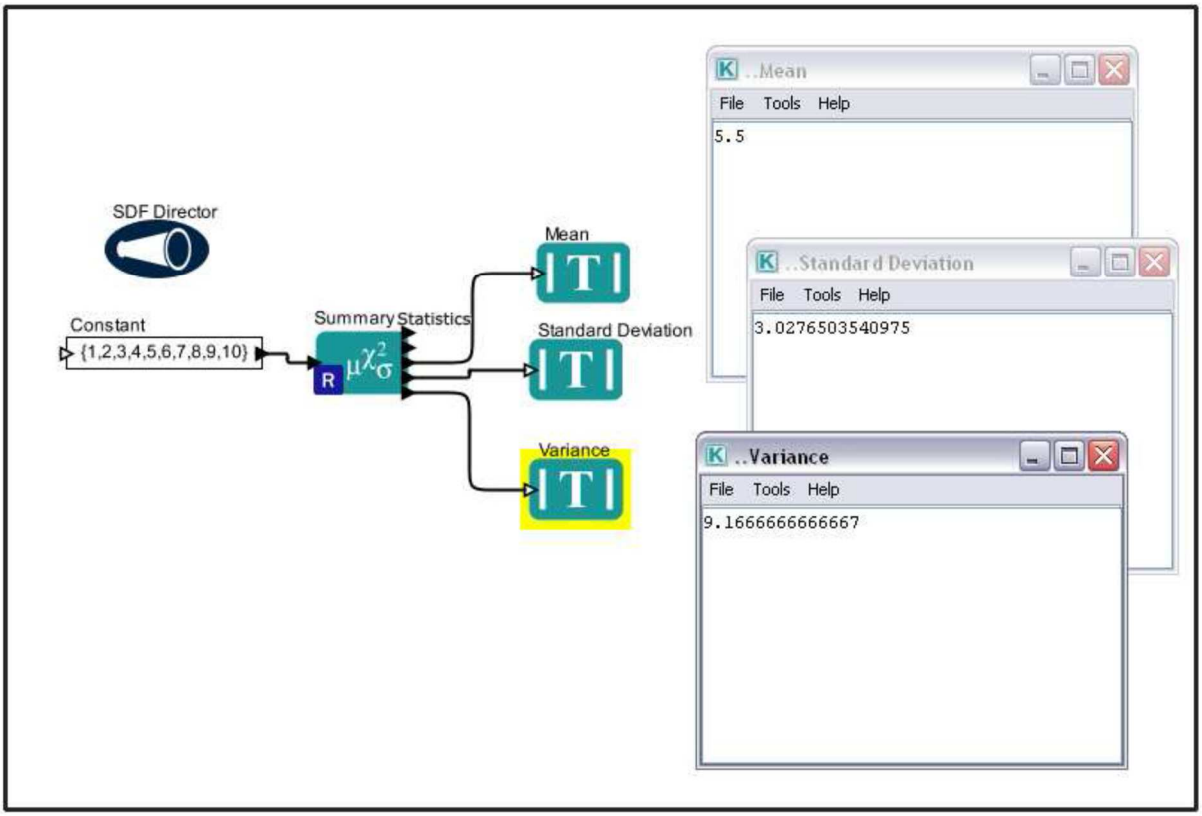


Figure 5. Create a simple workflow by Kepler (The kepler project, 2015)

The Kepler framework includes several actors for distribution sampling (Normal and Uniform) and statistical Analysis (e.g. Kendall, Pearson and Spearman correlation analysis). In addition, statistical analysis actors implement various graphical analysis methods (Bar chart, Scatter Plot) (Kepler Actor Reference, 2016).

3.5 Ecolego Software Package

Ecolego (Avila, Broed, & Pereira, 2003) is a MatLab toolbox that is primarily used for carrying out risk and safety assessments of dynamic systems and especially for the field of radiological risk assessment using deterministic or probabilistic simulations.

This software provides a database of radionuclides, their decay chains, and a parameter database. The Graphical user interface of Ecolego is equipped with different tools to support various modeling tasks. In Addition, the graphical user interface of Ecolego provide the ability to represent the dynamic models in forms of interaction matrices (see Figure 6) instead of traditional flow diagrams to facilitate the possibility to create and make complex and large models (Avila, Broed, & Pereira, 2003) (Broed & Xu, 2008). Using a graphical user interface, the conceptual models, designed using an interaction matrix, will be converted into mathematical models in Simulink (Avila, Broed, & Pereira, 2003).

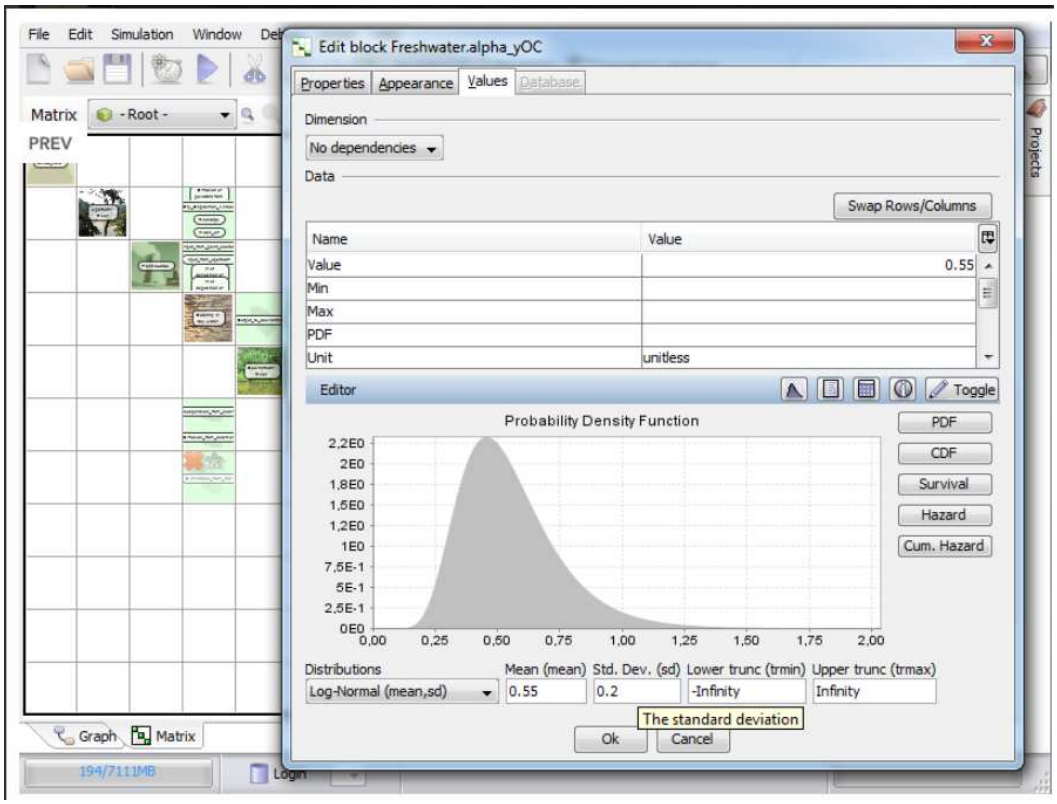


Figure 6. User Interface of th Ecolego (Facilia, 2016)

Probabilistic simulations are supported in Ecolego using Monte Carlo or Latin Hypercube sampling methods. Pearson and Spearman correlation coefficients as well as Sobol' indices like EASI (Plischke E. , 2010)are used in order to do Parameter sensitivity analysis in Ecolego (Facilia, 2016). Using the Eikos Sensitivity Analysis Toolbox (Facilia, 2016) adds the following sensitivity analysis methods to the Ecolego toolbox:

- Morris screening method
- Extended Fourier Amplitude Sensitivity Test (EFAST)
- Sobol' (first, custom and total order)
- Random balanced design
- Local sensitivity
- Garten's method

At the final step, the obtained results and outputs can be re-evaluated using post-processing functions, without rerunning simulations. Ecolego player is a free software package, which is capable of running models and performing the calculations without the need to access the Ecolego software itself. Once the models have been implemented in Ecolego, they can be distributed with the help of the Ecolego player. It should also be noted that the Ecolego player has all the functions of Ecolego but the structure of the created model cannot be changed in this tool.

The Ecolego toolbox utilizes the numerical solvers of MatLab in order to run the simulations. Furthermore, it can use MatLab or Java codes written by the user in the simulations (Facilia, 2016). Ecolego also uses the graphical tool of MatLab to present results (Avila, Broed, & Pereira, 2003).

3.6 SALOME Platform

SALOME is a generic platform that has been developed and supported by CEA (Commissariat à l'énergie atomique et aux énergies alternatives) and EDF(Électricité de France) Research and Development. It provides a modular architecture in order to perform pre- and post-processing for numerical simulations. Each module specifies a group of functionalities which are depicted in Figure 7. SALOME makes it possible to couple various codes to different scientific projects (Bergeaud & Tajchman, 2007) (Bergeaud & Lefebvre, 2010) although it is not equipped separately with numerical solvers. SALOME is a common tool used by CEA and EDF R&D in nuclear and industrial research fields. It has been developed to cover all steps of a study by providing an integrated platform to design simulation models, coupling simulation codes, data exchange, and managing the simulation steps and results.

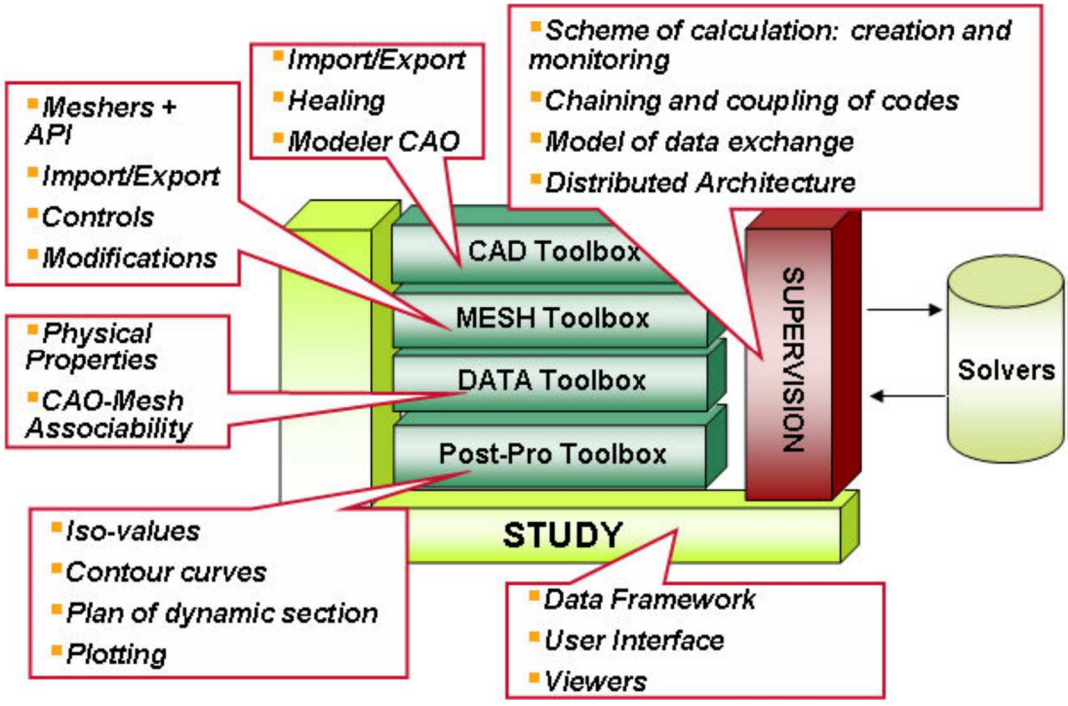


Figure 7. SALOME Architecture (CEA/DEN, EDF R&D, OPEN CASCADE, 2015)

The modular architecture and distributed computing ability of SALOME Platform make it a popular software tool to model, simulate and process in a variety of simulation contexts (Golfier, et al., 2009) (Cacuci, et al., 2006). The external code coupling (Bergeaud & Tajchman, 2007) in SALOME is based on a common data exchange model (Open Cascade. Salome6, 2012). This helps the different codes to interact with each other in a common way. It needs to integrate the external software with the SALOME Platform via wrapper generators with little modifications (Bergeaud & Tajchman, 2007). SALOME is able to manage distributed modules and solvers in form of a calculation workflow. (CEA; EDF; OPENCASCADE, 2016).

3.7 EMOS

EMOS (Buhmann, 2000) is a module based software solution, which started being developed in the middle of 1980's. It is used to carry out long time safety assessments for chemical toxic or radioactive waste repositories in any formation like salt, granite, etc. The modules contained in this tool make it possible to simulate the transport of contaminants in different components of the repository system and performing the probabilistic calculations. EMOS is also equipped with a simple graphical user interface (GUI), documentations and tools to work with data in tabular or graphical forms. Both deterministic and Monte-Carlo simulations of models and the post processing operations for

sensitivity analysis are supported in this software package. The modular design approach of the EMOS makes the design of models more flexible. It means a user can make his/her model with the help of modules, each of which is responsible for calculations of one component. The user is limited to use embedded modules and cannot involve external modules or couple new libraries to the EMOS tools. Adding new tools is not supported and a user should wait for new versions to extend his/her model with the new added modules to the toolkit.

3.8 ALLIANCES: Simulation Platform for Radioactive Waste Disposal

For the purpose of safety assessments of radioactive waste, CEA, ANDRA (Agence Nationale pour la Gestion des Déchets Radioactifs) and EDF developed a platform to simulate highly coupled thermo-hydro-mechanical-chemical- and radiological (THMCR) processes and simulate the evolution of radioactive waste storage and disposal facilities (OPEN CASCADE SAS, 2016). The first version of Alliances was released in 2003. Developing a new numeric tool is not the goal of Alliances. Alliances provides an integrated workspace to integrate legacy codes in the form of software components. It also provides a common data model for sharing data between coupled software components. This framework uses the SALOME Platform and its known format to perform pre- and post-processing tasks. Alliances has been designed under a multi-layer architecture. In this architecture, the legacy codes are categorized by their abilities of simulating different processes. The common data model is shared between these different layers of the Alliances architecture. Multi-level architecture helps to integrate new components and to connect the components with each other. User interfaces provided in textual and graphical modes allow a user to choose any component and arrange the solving strategy. The attached legacy codes must use the same API (Application Program Interface) in order to be usable in this framework. Included legacy codes are not directly accessible. They are linked to the Alliances platform as dynamic libraries.

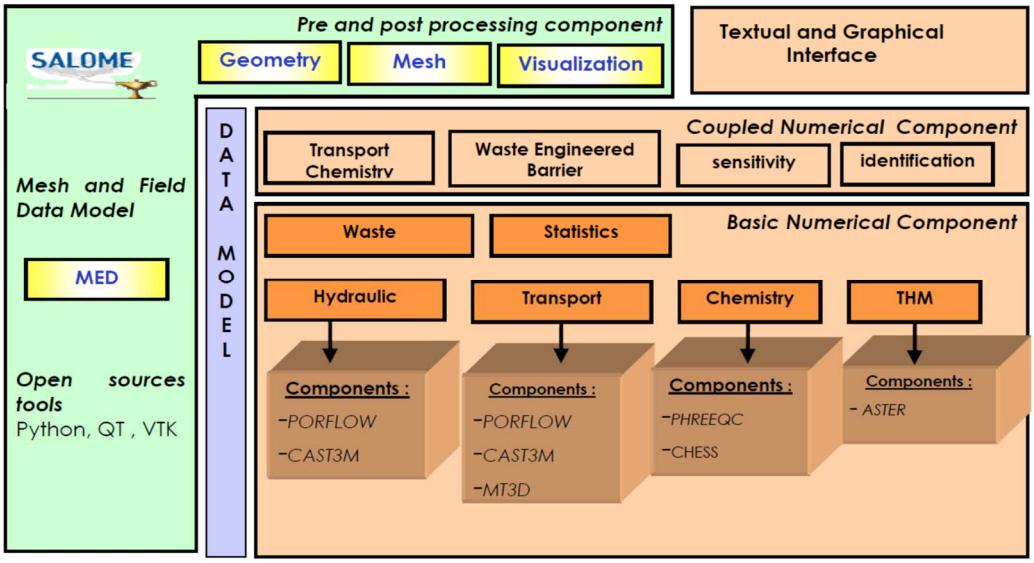


Figure 8. Multy-layer Architecture of Alliances platform (Deville, Montarnal, Loth, & Chavant, 2009)

The sampling module in Alliances provides Simple Random Sampling and Latin Hypercube Sampling methods in order to perform probabilistic simulations. The analyses tool of Alliances is able to calculate statistical indicators (moments, quantiles and curves) in order to perform uncertainty analyses. In addition, the sensitivity analysis is also supported by this tool via regressions (SRC/SRRC) and correlations (Pearson, spearman, PCC/ PRCC) coefficients (Montarnal, et al., 2006) (Deville, Montarnal, Loth, & Chavant, 2009).

The basic numerical components of Alliances can be selected for different simulation cases. In addition, the coupled numerical components include some predefined coupled numerical components based on basic numerical components (e.g. transport/chemistry). Thus, building new computational couplings between numerical components requires programming knowledge (Montarnal, et al., 2006).

The embedded modules and components of Alliances communicate with each other using the Data Model layer. Although this layer is a common channel to share the required information, it does not support the data interaction between coupled processes. The interactions between the coupled numerical components take place in their implementation level (Montarnal, et al., 2006).

3.9 AMBER: Compartment Modeling Tool

AMBER (Quintessa, 2015) is a MS-Windows compatible compartment modeling software (Punt, Smith, Herben, & Lloyd, 2005) developed by Quintessa. This software tool is specialized for building dynamic models, which simulate the migration and transfers of (decaying or non-decaying) contaminants in compartmental systems.

AMBER provides flexibility to build any case specific model and manipulate generic models to fulfill the requirements of model specifications (Quintessa, 2015). Reusing models as sub-models in other models can be considered as strength of this software. This feature can be used to model and simulate very complex systems.

The embedded solvers of AMBER (Quintessa, 2015)provide a widely applied software supporting assessments of model environmental systems and to calculate the doses due to long-term nuclide release (Lee Y. , Hwang, Kang, & Hahn, 2005) (Lee, Kang, & Hwang, 2007) (Lee M. , Hwang, Hyung, & Soo, 2005). This feature of AMBER is applied in safety assessment evaluations (Little, et al., 2009) (Lee Y. , Hwang, Kang, & Hahn, 2005).

Some important abilities of AMBER are as follows (Quintessa, 2015):

- AMBER addresses uncertainties in model, scenario and parameters
- The visual front-end helps the user to make an illustration of how the model components are distributed in a model space
- The Monte Carlo and Latin Hypercube Sampling tools help to generate samples for the input of simulations with selecting a defined probability density function
- Different graph types (Line, Scatter, CDF (Cumulative Distribution Function), etc.) to report the result of probabilistic simulations
- Export of results to external file formats

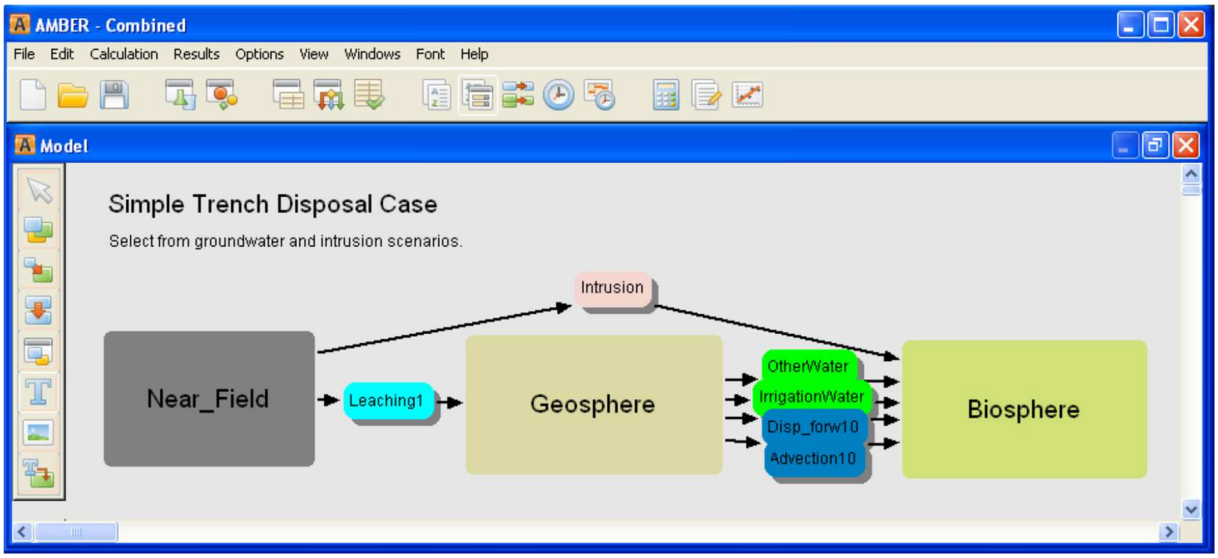


Figure 9. Amber modeling tool (Near surface disposal of llw, 2015)

The big advantage of AMBER is its flexibility and ability to tailor models for their specific site issues.

3.10 GoldSim

The goal of the GoldSim (Kossik & Miller, 2009) package is to design a general-purpose graphical simulation framework. GoldSim provides a probabilistic simulation framework to perform risk assessments of complex systems by Monte Carlo simulation (GoldSim Technology Group , 2016). The main features of GoldSim include the possibility to create and manipulate data and equations in a visual way. The multi-layer architecture also brings the capability to link external programs to the GoldSim model in a dynamic manner. Each simulated model can be executed either in a deterministic or a probabilistic way. This platform runs under the MS-Windows operating system.

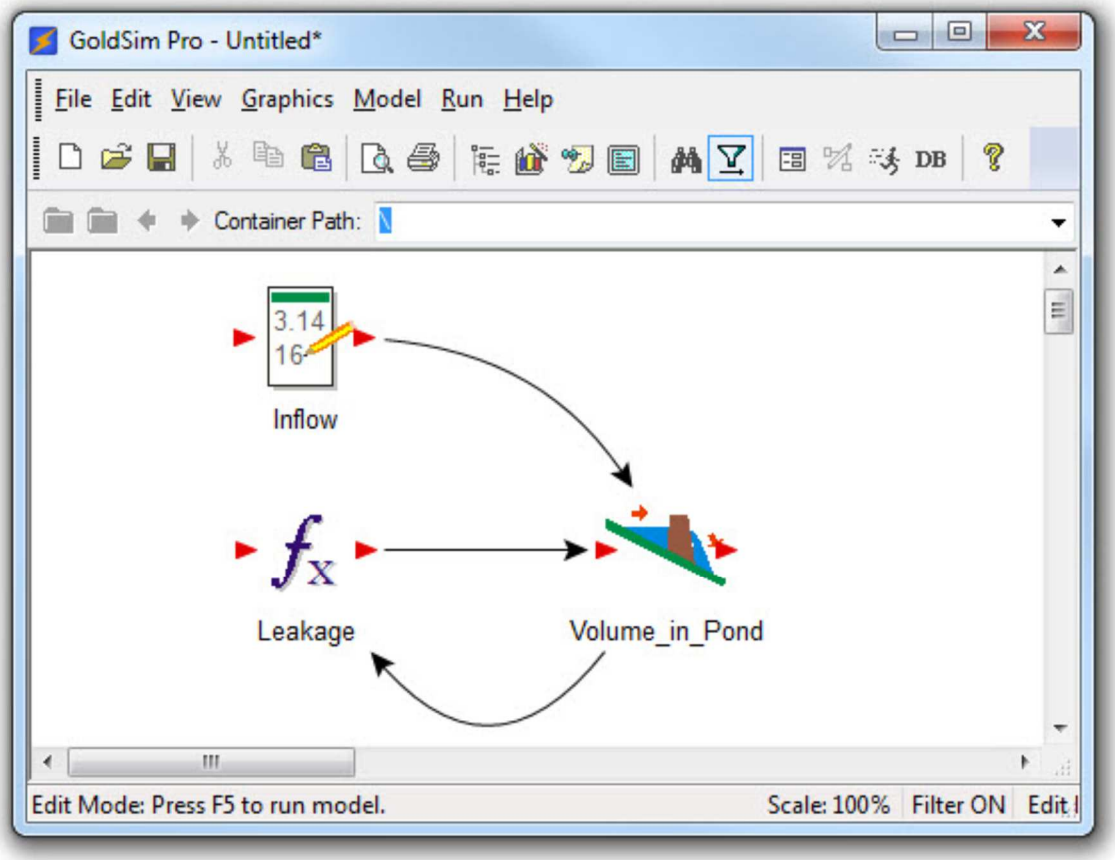


Figure 10. The GoldSim simulation Environment (GoldSim Technology Group, 2016)

Simulation models in GoldSim consist of elements and links. There are two types of elements: input entry elements and function elements. The user can define the input of the models by input elements and also write and consequently calculate expressions or run complex scripts with the help of function elements. The input and output of elements can be connected via links.

All the simulations are performed in a way that they enable the possibility for a user to identify, analyze and understand system controlling factors and additionally to predict the system state and behavior in any given state of time in future.

The probabilistic simulation in GoldSim can be started by defining some inputs of a model as uncertain and assigning probability distributions (so-called stochastics) to these parameters. GoldSim supports Uncertainty and Sensitivity Analysis, the latter in form of linear (Pearson) and non-linear (Spearman) correlation coefficients, coefficient of determination, SRC (standardized regression coefficient), Partial Correlation Coefficient and Importance Measure (Sobol' first order effect) (GoldSim Technology Group, 2006).

GoldSim is used as visual spreadsheet tool to create setup, manipulate and solve equations. The equations in GoldSim are represented by the objects and their interdependencies. The equations of GoldSim integrate all of the various sub-models into a single total system model in a hierarchical and modular manner. As a dynamic simulator, GoldSim models can evolve and change over time. GoldSim input elements can be imported as data sources in the form of spreadsheet tables and may be linked directly to an Open Database Connectivity (ODBC) compliant database.

Another main feature of GoldSim is integrating differentiated scenarios. Differentiated scenarios bring the possibility to optimize a model, ask logical *What if* questions, perform sensitivity analysis, test and compare alternative designs. The obtained results from the model can be effectively analyzed,

documented and presented in various forms such as charts, diagrams and tables with the use of built-in features of GoldSim such as charting tool, model documenting tool and GoldSim Player.

Highly specialized extension modules are also available to be used alongside the GoldSim simulator. Some examples of these extensions are as follows (GoldSim Technology Group, 2016):

1. Financial Module
2. Contaminant transport Module: Specialized elements for representing contaminant species, transport media, transport pathways, contaminant sources and receptors, and the coupled sets of differential equations underlying these systems
3. Reliability Module
4. Dashboard authoring Module
5. Distributed Processing Module: Multiple realizations of a system are simulated. Distributed simulation on different computers in a network

Additional modules can be also used by GoldSim as compiled DLLs by developing a wrapper (shell) around the existing functions. In this case, data are passed from GoldSim to the external function and back again to GoldSim via arrays of double precision floating point input and output arguments. GoldSim is applied in safety assessments related to HLW repositories (Lee & Hwang, 2009)and used for comparisons with Amber (Punt, Smith, Herben, & Lloyd, 2005).

3.11 Simulink

Simulink (Simulation and model-based design, 2015) is an extension package of the MatLab (MathWorks, 2016)software. Modeling, simulating and analyzing dynamical systems (Abramov, Mannan, & Durieux, 2009) are the main goals of this software. The Graphical User Interface helps a user to put the drag-and-drop model parts together and rapidly build the desired models. The hierarchical design of models and model components are practical features of this modeling tool. Simulink has a comprehensive library of model blocks, components and connectors. The library components can be extended by user defined and customized blocks. Simulink converts block diagrams to a mathematical model and predicts the result of the model using different solvers. Simulink models and their components are accessible during their runtime to check the values from Simulink menus or the command line of MatLab. The command line window is also helpful for running a batch of models and simulations. The interaction between MatLab and Simulink is useful for pre- and post-processing procedures. The input and output values of the simulations in Simulink can be stored in the MatLab workspace to analyze and use in further use cases. Managing big projects, generating C/C++ codes from projects and connecting hardware are other features of the Simulink tool.

3.12 ReSUS (Preceding Version)

ReSUS is the Acronym for **Repository Simulation Uncertainty propagation and Sensitivity analysis**. The ReSUS project came into existence at the Institute of Disposal Research at Clausthal University of Technology. The purpose of this project is to develop a flexible software platform for the probabilistic analysis in the frame of safety assessments for radioactive waste repositories. It has been developed to achieve the ability of coupling and coordinating the execution of different external executable simulation programs. The main point of this approach is to utilize the template based method for uncertainty propagation and sensitivity analysis. The impact of input parameters on simulated model outputs can be studied by the probabilistic execution of models in ReSUS.

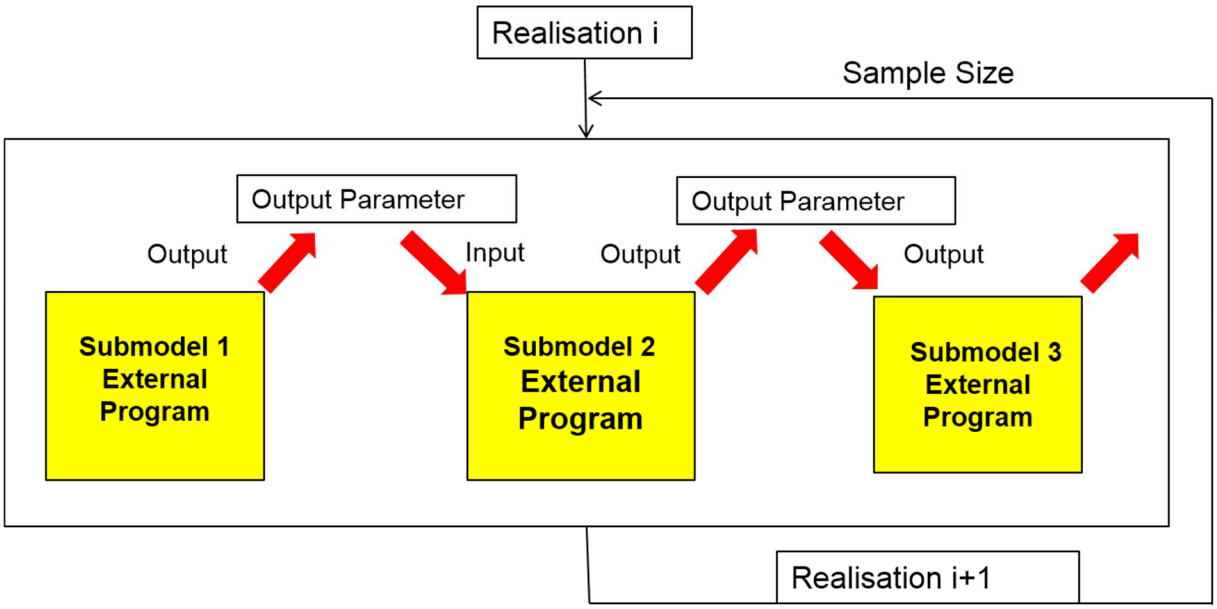


Figure 11. The simulation mechanism of preceding version of ReSUS (Li, 2015)

The working mechanism of the ReSUS is based on automatic replacements of parameter values in a template file and it generates the proper input for each run of simulation on model. The template based approach can also be used to transfer the outputs of one model to the input files of another model. This means that the places (placeholders) for parameters in a template file can be filled with one or more values. In most cases, the time dependent values will be transferred between sub-models. The output values of one sub-model can be transferred to a new model and be used as boundary conditions, source points, material parameters, etc. in their input files (Li, 2015). Some programming knowledge is needed to couple new external programs to this version of ReSUS. A user can connect the coupled programs to make a model chain. A Graphical User Interface (GUI) is available to do the simulations and analysis. Handling run-time errors is not supported in this version of ReSUS Platform.

3.13 State of the Art: Overview and Summary

In this section, the methods and software tools which are developed to provide a probabilistic framework in order to perform uncertainty and sensitivity analysis on simple and complex models are investigated. The summary of the introduced software and their supported aspects and capabilities is listed in Table 1. At first view, this table shows the aspects and features related to each tool in a row. It further shows that some of functionalities such as probabilistic simulation and post-processing operations are supported by almost each simulation tool, however none of these tools offers all the presented features in an integrated way. These tools are developed for solving special problems or simulating certain aspects of models, and none of them can be used as an integrated platform for probabilistic simulations. They are not usable for wide variety of problems. Most of them don't support the using external simulation programs in the probabilistic simulations and sensitivity analysis.

The aim of this overview is to give better understanding of the problem that we want to solve in this thesis. Consequently, knowing the methods and tools which are used to solve the existing problem in other projects is helpful to a better solution.

Table 1. List of features and abilities of introduced simulation tools

Methods	Probabilistic	Sensitivity Analysis	Workflow Management	Debug	Using DLLs	Assigning External Executable	Using External Solvers	Used as Library	ASCII Templates	Parallelization Supported
JRC SimLab	✓	✓		✓				✓		
DAKOTA	✓	✓		✓				✓	✓	✓
OpenTURNS	✓	✓						✓		✓
Kepler	✓	✓	✓	✓		✓				
Ecolego	✓	✓	✓					✓		
SALOME	✓	✓	✓	✓	✓		✓			✓
EMOS	✓	✓	✓				✓			
ALLIANCES	✓	✓								
AMBER	✓	✓	✓				✓			
GoldSim	✓	✓	✓		✓					✓
MatLab Simulink			✓	✓			✓			
ReSUS Starter	✓	✓				✓			✓	

As mentioned in Section 2, the probabilistic simulation and sensitivity analysis are the key features of the software platforms which are used to perform safety assessments. The two first columns of Table 1 also show that almost every simulation tool supports these capabilities. In order to apply these methods in more complicated simulations, the workflow management method has to be integrated into probabilistic simulations. However this method is not fully supported by these simulations platforms (see the third column; workflow management).

The fourth column of Table 1considers the Debugging option and shows that this capability is not fully supported by every simulation tool. Debugging allows users to find and fix the possible problems in the simulation process. This option can improve the reliability of the simulations.

The software tools listed in Table 1include a variety of abilities; however, they mostly do not address two particular aspects: The ability of using external programs is limited to supporting DLLs as extensions for adding some functionalities or new modules in SALOME and GoldSim. It is also clear that binding DLLs requires additional programming knowledge. On the other side, assigning the third party executables to simulator platforms as extension is also not supported by the presented

software platforms. Some of these platforms which are investigated in this section are able to use the external computational components by implementing some interfaces for connecting to the external solvers. The disadvantage of this method is that it is confined to predefined interfaces that are utilized for connecting the external solvers to the simulation platforms. This implies that if the external solver does not support the predefined interface, it is not possible to use the functionalities of the external solver in the simulation platform.

Using the ASCII based template files for preforming probabilistic simulations is only supported by DAKOTA and the preceding version of ReSUS to a limited extent. ReSUS uses this method only for certain predefined simulations. Using the Template files in Dakota is utilized for generating input files with its additional pre-processing tool.

A category of presented platforms has the capability to be included as a library in other projects (e.g. the DLL files of the JRC SimLab can be assigned to C++ projects).This property enables the third party developers to utilize the functionalities of such platforms in their projects. However such capabilities are restricted to only four of the presented software tools.

Parallelization is also an additional option that is considered in the Table 1 which is supported by only a few tools. Parallelization means the ability to concurrently run the simulations components in order to better use the available resources and to improve the speed of the simulations.

4 Motivation

The overview of the state of the art in Chapter 3.13 shows that most software tools, which are developed to support the probabilistic simulations and perform uncertainty and sensitivity analysis, use their embedded tools and solvers. Furthermore, some of the introduced tools apply the workflow management concepts to the probabilistic simulations in order to simulate the multi-layered models and systems which are particularly suitable for simulating the migration of the radioactive particles through layers of a repository system. The solvers and simulation components in such platforms support just special cases, which are already embedded in their platform. This feature imposes restrictions to the flexibilities of modeling and simulation processes.

From one point of view, the investigated software tools are trying to extend their functionality with the help of adopting the external tools in the form of assigning Dynamic Linked Libraries (DLLs) or using external solvers from other platforms. The problem arises since this method is not fully supported by these tools and on the other hand, using such facilities is limited to the architecture of the simulators and requires sophisticated programming knowledge. It means the provided methods are not generally usable and flexible. Furthermore, these methods can only use embedded solvers which are already defined for special models and processes. It means, if the intended problem is changed then it is not guaranteed that the simulation or solver program can still support the modeling, simulation and analyses of the phenomena.

From another point of view, there is a high and increasing number of special-purpose simulators, e.g. for modeling specific, often coupled THMC problems, which are developed by third party developers to support various processes in variant types of models and which are not included in any of the probabilistic simulation tools. The deterministic manner of such external third party simulators and difficulties by accessing their source codes makes it problematic to utilize their power to do the probabilistic simulations by the existing platforms.

The development of the ReSUS Platform was initialized with the objective of overcoming the aforementioned lack of the capability and flexibility in the existing probabilistic simulation platforms. The objective of this effort is to develop a simulation platform which includes the basic functionalities (e.g. sample generation, probabilistic simulation, probabilistic analysis) of the current probabilistic simulation platforms and in addition provides new generic capabilities to cover their missing features and to resolve their architecture constrains in working with the third party simulation tools.

5 Objective

The objective of this dissertation is to overcome this lack of compatibility in the probabilistic simulation platforms which is mentioned in the previous section. This approach focuses on providing a flexible way to assign and manage the third party deterministic simulation programs in probabilistic simulations while the basic common functionalities of existing tools are supported.

The following points should be considered as essential functionalities and features of the platform the development of which is described in this thesis:

1- Providing an integrated platform which supports the whole process of probabilistic simulations in addition to pre- and post-processing operations

   a. Designing the probabilistic simulation models with a graphical aided tool which allows creating and manipulating the simulation scenarios

   b. Sampling input values for simulation models with different sample generator methods and probability distributions

   c. Propagating the uncertainty of input values to the simulation models during repetitive runs of simulations

   d. calculating the sensitivity measures

   e. illustrating the input and output values of the simulations in visual and tabular form

2- Utilizing the power of the third party simulation programs, in particular the deterministic simulation tools, in the probabilistic frame

3- A generic approach that has the possibility to get used in any field of research and study

4- Using the flexibility of the workflow of processes to make chains of simulation tools in order to model and simulate the simple and complicated systems as well as managing the process and data sequence automatically

5- Ability to import/export the simulation input/output values from/into the third party Analyzer Tools

6- Providing the debugging capability to easily find and fix of the simulation model failures

6 Software Development

A Software development process is a structure to describe the steps required to produce a software tool. Most models of the Software Development process are based on the following common steps:

1- Requirement analysis

2- Design

3- Implementation

4- Testing

5- Evaluation

The process to develop the ReSUS Platform also includes these steps. They are used in an iterative way to produce the ReSUS Software Platform. This thesis avoids explaining the details of the whole process in every iteration and only introduces the significant concepts and steps that are used for developing the ReSUS Platform. In this chapter, the design issues of the ReSUS Platform including their requirement analysis are considered. The rest of the development process steps will be described in the next chapters.

6.1 Requirement analysis

The requirement analysis of this project consisted of a series of discussions among the members of the development team. The content of these discussions involved the state of the art, as described in the previous section (in particular the preceding version of the ReSUS (Li, 2015)), the current state of the project, the prototypes and feedbacks from the other members, and the proposals about how to proceed in the future. The discussions were scheduled for once a month.

The information gathered and the analyzed via these meetings and cooperating with other members of the group helped us to extract the functional requirements of the software and to develop a prototype product. For more details, please refer to the developer handbook of the ReSUS Platform (Ghofrani J. , 2016).

6.1.1 Product Perspective

This platform is developed as a replacement for the preceding version of ReSUS (Li, 2015) . It implements the functionalities provided by the preceding version of ReSUS and concurrently improves the flexibility of this platform by providing new functionalities. The significant functionalities aimed at in the current version comprise a flexible graphical designer with the capability of building workflows and the ability of adopting third party executables without the need of manipulating the source code of ReSUS.

Figure 12 illustrates the three major components of ReSUS and their communication which is based on an XML file. It also shows that the User Interface is used to assign external third party simulator programs to the Core. Furthermore, one can see that the results of probabilistic simulations collected by the Core are stored in binary files which are supported by the Analyzer Tool.

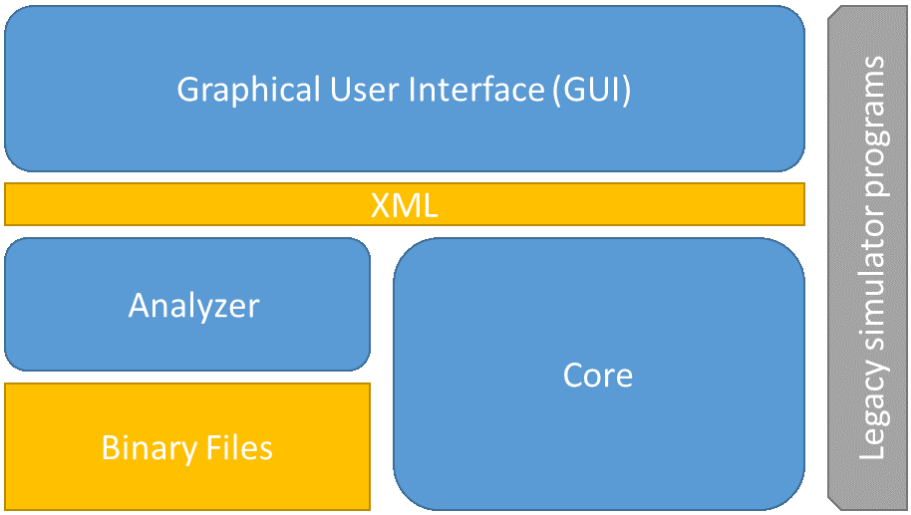


Figure 12. Architecure of the ReSUS Platform

6.1.2 Documentation

In addition to the current document, a user handbook (Ghofrani & Li, 2016) and a developer handbook (Ghofrani J. , 2016) are available. All the simulation components, their properties and issues which are necessary to utilize the functionalities of the ReSUS Platform are described in the user handbook. The Details of the development process including analyzing the requirements, design, development, test and deployment issues of the ReSUS Platform are available in developer handbook.

6.1.3 Product Functions

The major functionalities which the ReSUS platform should provide are listed below:

1- Generating random samples for probabilistic parameters of the simulations

2- Utilizing third party external simulation programs for probabilistic simulations

3- Designing and Preparing the probabilistic simulation workflows

4- Performing and managing the probabilistic simulations

5- Performing Uncertainty analysis

6- Performing Sensitivity analysis

7- Providing an interface for connection to other probabilistic tools

6.1.4 Data Model

The data models used in order to organize the simulation tasks, inputs and outputs are described in this subsection. It gives the user a brief overview of the data structure used by ReSUS.

The pre- and post-simulation tasks are defined as XML files. The Graphical User Interface is capable to convert the user requests and task definitions into the XML formatted files. These files are readable by the Core and the Analyzer Tool components.

The simulation input files will be stored in HDF5 formatted files (Folk, Heber, Koziol, Pourmal, & Robinson, 2014). GUI allows the user to generate and explore these files and the simulation results (input and output values) will be stored in binary files which are used by the Analyzer Tool.

Figure 13 illustrates the data files which are used as communication media between the basic components of the ReSUS Platform. In Figure 13, The arrow from XML file into the GUI shows that the

GUI can read and modify XML files which contain the simulation tasks and configuration. These files can be read and interpreted by the Core component. In addition, the GUI can initialize the configurations of the post-processing tasks in the form of XML files which are performable by the Analyzer Tool. The inputs of the simulations are stored in HDF5 files which will be read by the Core during the simulation run-time and the Result files of the simulation will be stored in binary files. These binary files are interpretable by the Analyzer Tool.

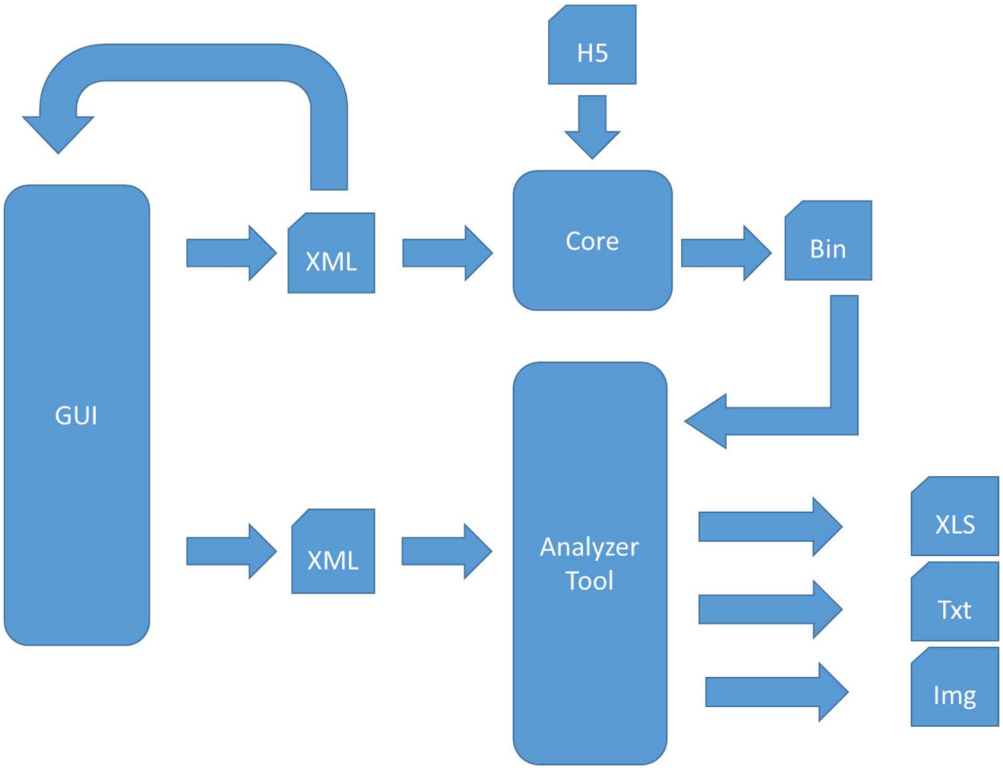


Figure 13. Used Data between Components of ReSUS

The HDF5 files keep the values of input parameters of the probabilistic simulations. For each defined probabilistic parameter in a HDF5 file, there is a dataset with N values, with N specifying the number of the simulation runs (realizations).

6.1.5 User Interfaces

The ReSUS Platform provides three interfaces to interact with the users; Graphical Model Designer, Chart Tool and Export Tool.

Upon starting ReSUS, the Graphical Model Designer appears (Figure 14). The user will be able to prepare a simulation workflow with the available graphical tools in the Graphical Model Designer.

The Graphical Model Designer allows the user to specify the required steps to make a probabilistic simulation which include:

1- Specifying the workflow of the simulation
2- Specifying the deterministic parameters of simulation
3- Specifying the probabilistic parameters of a simulation together with their probability distributions
4- Generating samples for the probabilistic parameters according to the specified probability distributions
5- Assigning the external simulator programs

6- Configuring the input, execution and output of the external simulators
7- Configuring the automatization of the gathering and storing the probabilistic simulation results
8- Starting the simulation
9- Starting the Analyzer Tool

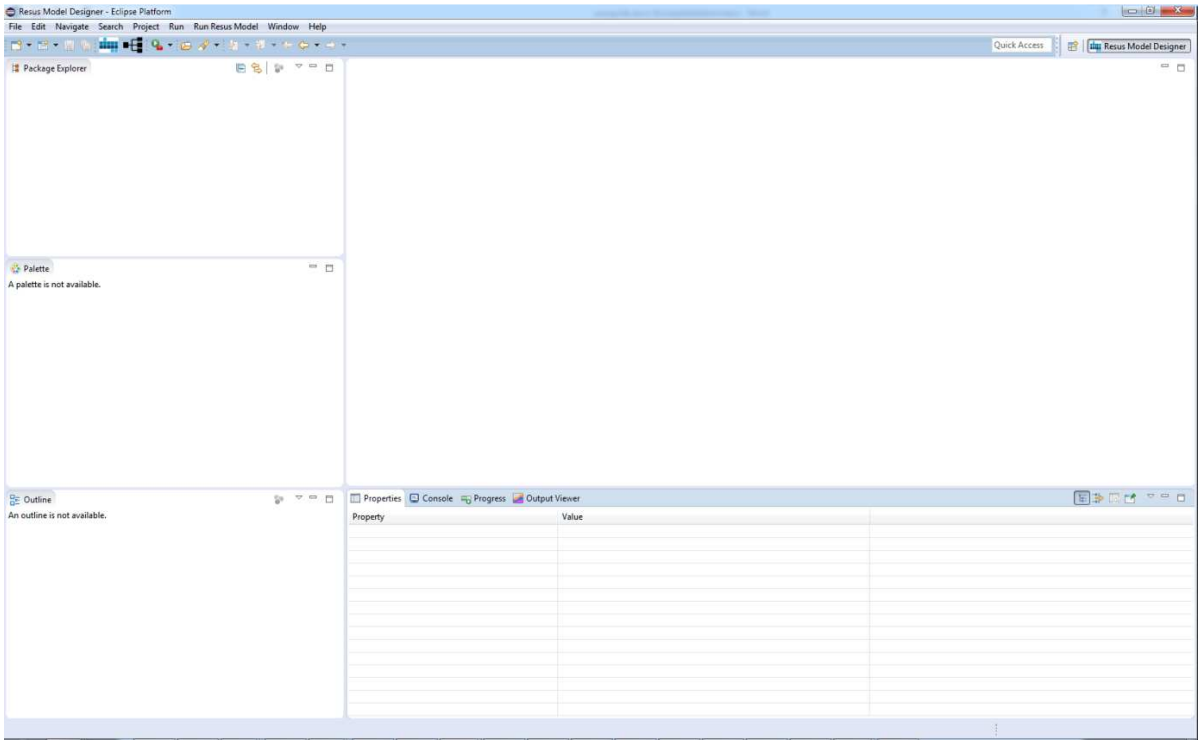


Figure 14. ReSUS Model Designer

This interface offers three main graphical components to build the probabilistic simulation workflows (Figure 15) with simple drag and drop possibilities. The Input Provider components (red triangle in Figure 15) are used to specify the starting point of the workflows. The Model Frames (blue box in Figure 15) provide an interface to assign the external simulators into the workflow and to manage them. The Result Converter (black triangle in Figure 15) is responsible to extract the simulation results from the output file of the simulator and store them in a binary file for post-processing objectives.

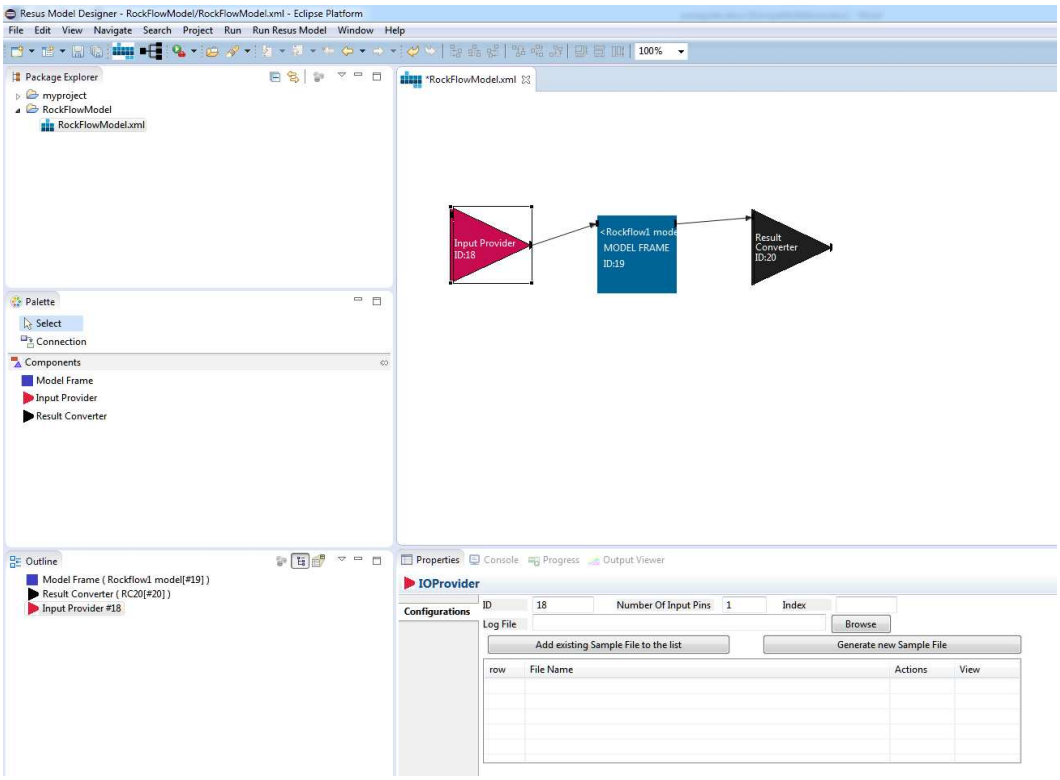


Figure 15. Defining a probabilistic simulation workflow in the ReSUS Graphical Model Designer

The starting point of the simulation workflow should be defined with an Input Provider component while the end point of the simulation is specified by a Result Converter and a Model Frame is the middle component. Each of these graphical components have at least one input and one output pin on their left and right side, respectively, as small black squares. The pins specify the connection points of these graphical components with the Connection components (see below). Figure 16 illustrates a simple simulation workflow.

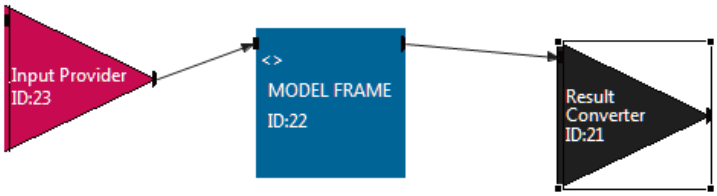


Figure 16. A simple simulation workflow

The simulation workflows can be connected to each other by means of Connection components (arrows in Figure 16) to build complicated simulation workflows in order to represent the multilayered systems. An example is shown in Figure 17.

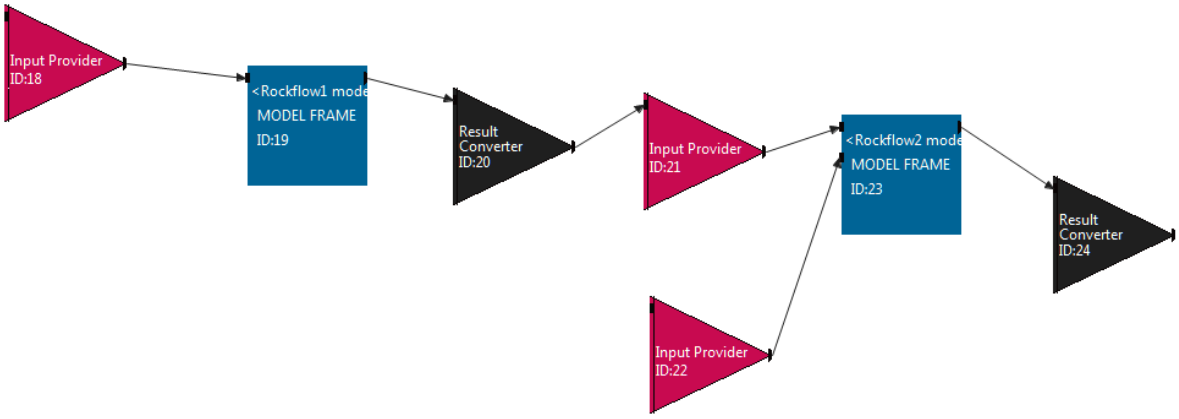


Figure 17. Building complicated models by connecting simple workflows to each other

ReSUS utilizes ASCII based text files to generate the input files for the simulator programs. The template files specify the deterministic structure of the input file. Furthermore, it specifies the places of the probabilistic parameter values in the body of the input file of the simulator by placeholder signs (shown as red strings in Figure 21). The ReSUS Platform generates input files for the simulator according to their defined template file in their corresponding Model Frame. Figure 21 illustrates the schematic working mechanism of the Model Frame component to generate input files from template files using templates.

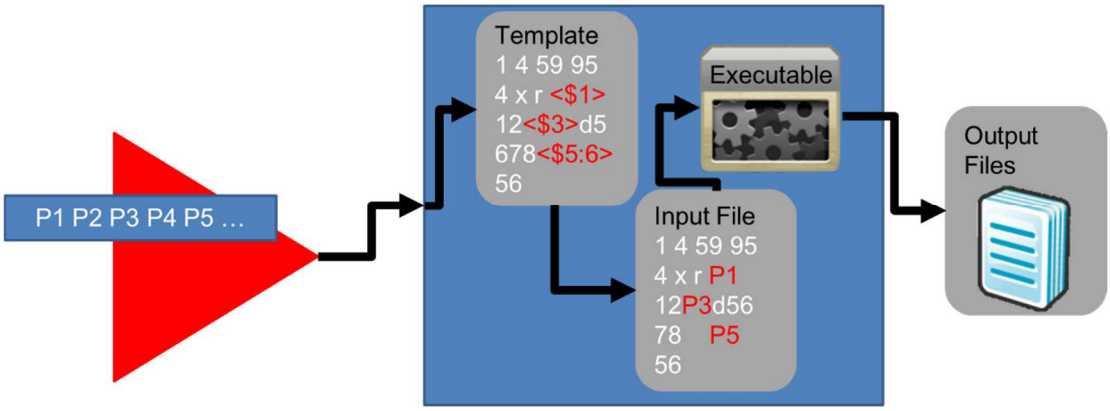


Figure 18. Schematic work of the Template files

The Random Generator tool, which is embedded in the GUI of ReSUS, generates random values for the input parameters of the probabilistic simulations. The values generated for each Parameter will be stored in a table with the name of parameters within an HDF5 formatted file. The Input Provider components are responsible to transfer the values from HDF5 formatted files into the Model Frames.

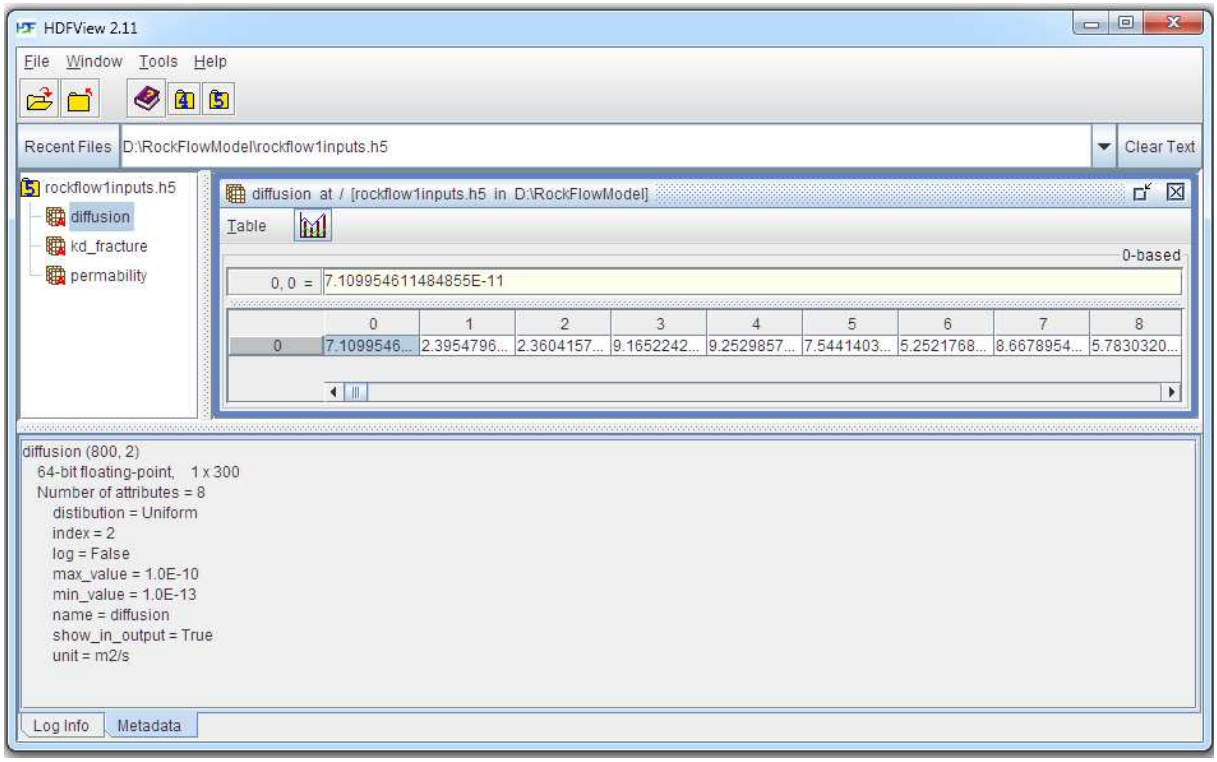


Figure 19. Sample values generated for each parameter as a dataset within a HDF5 file

The Result Converter provides the functionality to find certain structures in the output files of the simulator programs. It will be done with the help of Regular Expressions. Additionally, the Result Converter can split the text structure found into sub-strings using defined delimiters. The specified indices of the sub-strings will be stored in a structured binary file for post-processing operations and also transferred to through the simulation workflow if some Input Provider is connected to the Result Converter component. Figure 20 depicts the simplified working schema of the Result Converter component.

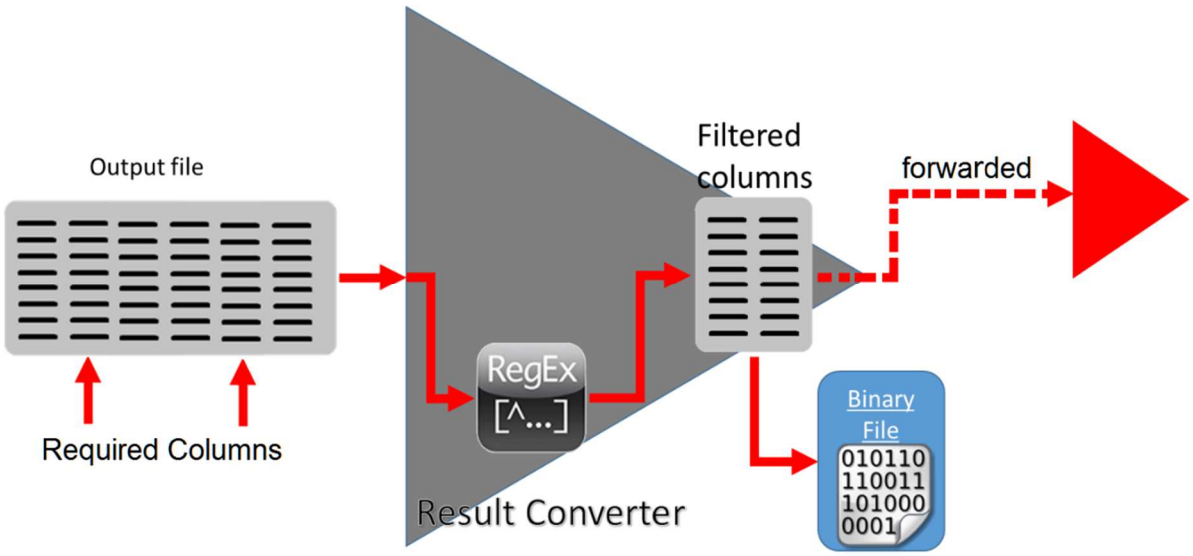


Figure 20. Working mechanism of the Result Converter

Furthermore, the ReSUS Platform provides post-processing functionalities via analyzing interfaces (depicted in Figure 21). Once the simulation is finished, the user will be able to visualize the results of

the simulation in various forms. The probabilistic simulation inputs and outputs will be stored in a binary file. The Analyzer Tool assists users to open the binary files and visualize their contents in the form of charts or tables. The Chart Tool is responsible to convert the binary files into charts. The Export Tool converts the binary files into tabular forms. Moreover, the user calculates the uncertainty and sensitivity analysis measures on the simulation results which are presented in both Chart and Export Tools. The user will be able to convert the content of the binary files into third party formats like CSV and XLS.

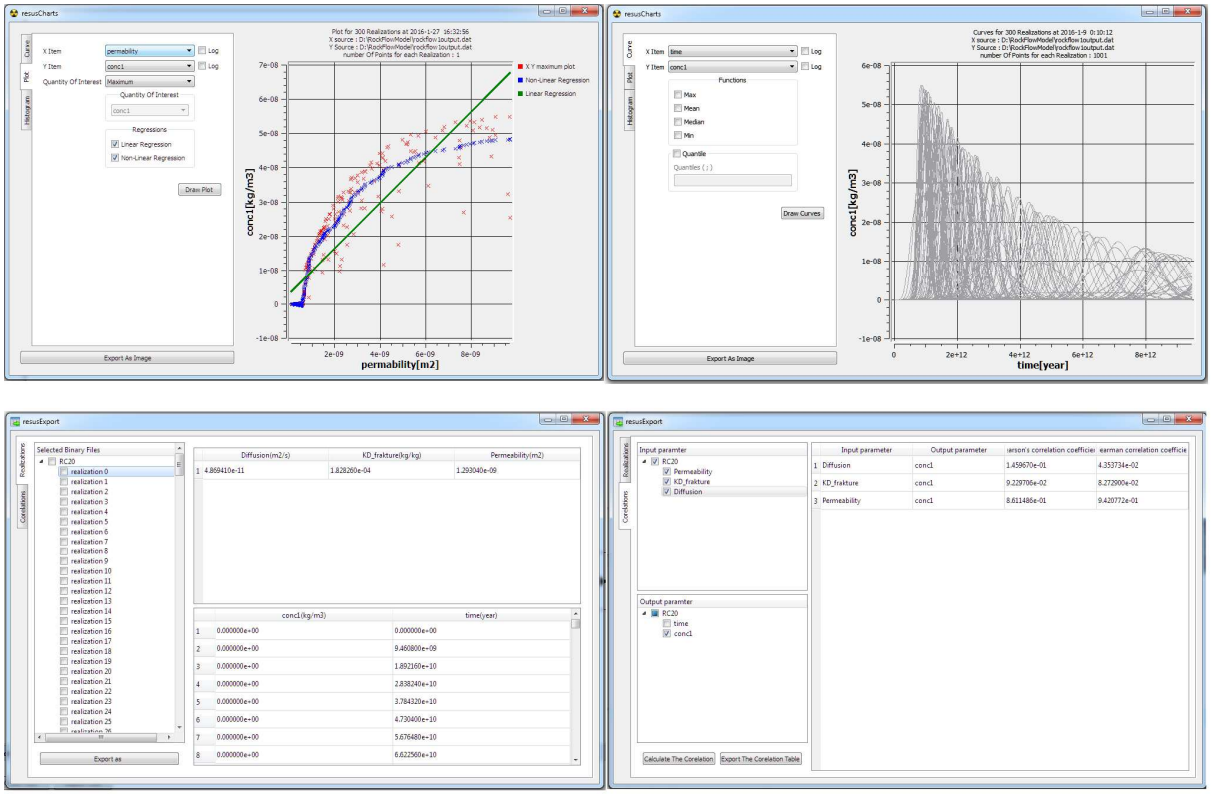


Figure 21. Visualizing the simulation results using components of the Analyzer Tool: Chart Tool (top) and Export Tool (down)

6.1.6 Summary

The user requirements which are considered in prototyping and further development of the ReSUS Platform are introduced in this subsection. The presented features and usage issues of this platform are used to show the view point of the user on this software tool.

6.2 Design

This subsection introduces the design issues of the software which were taken into account when implementing the ReSUS Platform. These issues describe the system from the point of view of the developer. More details of the entities and components introduced involving the description of their attributes and functions are available in the developer handbook (Ghofrani J. , 2016).

6.2.1 Architecture and Component Design

The Architecture of the ReSUS Platform is based on different layers. Each layer describes the logical categorization of components and their functionalities. The XML files take the responsibility of providing a common medium for interactions between layers. The complexity of methods and structures of each layer is hidden from the sight of the user and represented by other layers behind

the boundaries of that layer. Extending the architecture of the ReSUS Platform is also possible with this kind of design.

The design architecture of the ReSUS Platform comprises the three following layers:

1- Graphical User Interface (GUI): The GUI provides an interface for the ReSUS Platform to interact with users. The connection between the user and the other layers of the ReSUS Platform will be provided via this layer. This layer supports all required pre-processing operations and tasks to start a probabilistic simulation. This includes the functionalities with which the user can define the simulation tasks and their related details. These details consist of designing a probabilistic simulation and generating random values. Starting the simulation and preparing the communication messages to send to the other layers are also included in the responsibilities of this layer.

2- Core layer: This layer is responsible for processing and performing the simulation tasks defined by the user. This task includes assigning the solvers to the simulations, managing and synchronizing the simulation and dataflow.

3- Analyze layer: This layer provides the functionalities to perform the post-processing operations. These operations include presenting the simulation data in tabular and visualized form in order to perform uncertainty and sensitivity analysis.

Since the XML files are readable by humans too, the third party software and human agents can use the provided functions in the component of the ReSUS Platform by formulating their intended tasks and requests in XML format.

Despite improving the portability and flexibility of system by using XML files, the XML format is not suitable for storing big amounts of numerical values because of performance and volume issues. Therefore, the simulation results will be stored in the binary format.

The layers presented in this part use some entities to implement their responsibilities. In the following the entities of each layer are presented.

6.2.2 Graphical User Interface

The following entities are designed to implement the functionalities of the Graphical User interface:

1- Graphical Editor: This a graphical tool which helps the user to define the pre-processing operations such as creating or editing the simulation workflow as well as manipulating the non-graphical properties of the workflow members. This item is able to save and load the simulation tasks and configurations as XML files. It parses the XML files and initializes the graphical items according to the defined configurations in the XML nodes in an internal list of the graphical components. There are four graphical items which hold the information and configuration of the simulation components: Input Provider, Model Frame, Result Converter and Connection. Graphical Editor draws these items according to their graphical properties such as location and size. These graphical items are equipped with input and output pins which make it possible to create the connections between them.

2- Properties View: The non-graphical properties of the workflow building block items can be viewed and manipulated by the Graphical editor in the provided Properties View. It provides textboxes and tables to view and update the additional configurations of each graphical component

3- XML Editor: This is an embedded editor with the ability of showing the XML files in hierarchical format. This editor is able to view the content of the XML files in plain text.

4- Random sample generator: This is a program embedded in the Graphical User Interface which generates randomly sampled values according to various probability distributions (Uniform, Gauss, Gauss Triangular) and sampling methods (simple random and Low discrepancy sequence / Quasi Monte Carlo sampler (Sobol' LPtau)). The randomly generated values will be stored in a HDF5 file and can be viewed by the HDF Viewer tool from the HDF Group, as well as by the Sample Viewer tool of ReSUS. Moreover, this component is capable to generat, edit, save and load the configurations (Parameter) for generating the sample values.

5- Sample Viewer: This is a simple component of the Graphical User Interface which is able to load and view the HDF5 files which are created based on the format of the sample files of ReSUS.

6- Parameter: A unit to categorize and to describe the defined configuration for a group of values. These configurations will be used by the Random Sample Generator.

7- Template editor: A tiny helping text editor which is equipped with syntax highlighting capabilities. It makes it easier to work with placeholders in template files

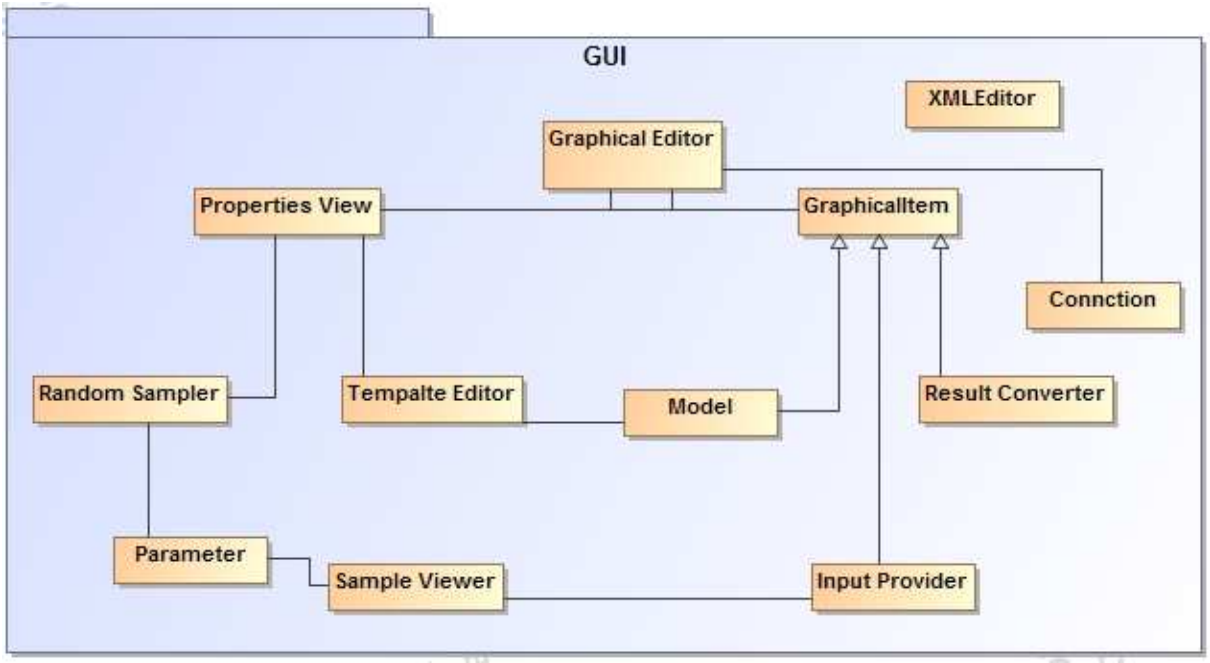


Figure 22. Simple Class Diagram of the GUI

6.2.3 Core

The Core Component consists of the following entities and concepts:

1- Simulator: The Simulator converts the simulation structure and configuration from XML into the Core framework components. Moreover, it manages and coordinates the execution process of the executable assigned to the probabilistic simulation.

2- Model: Model provides an interface to interact between the ReSUS Platform and the third party executables which are assigned to the simulation workflows. It provides inputs files, calls the simulator and check the output files generated by the executable. In addition,

managing the execution errors and controlling the simulator output files are included in the functionalities of the component.

3- OutputPair: OutputPair holds the configuration (min size and number of lines) for the consistency check after each run of the simulation which can be performed by Model to validate the size of the output files.

4- Input provider: It provides the interface between the stored randomly generated sample values (in a HDF5 formatted file) and the Model item. It provides the parameter values for generating the input file of the executable by Model. Furthermore, this component transfers the data between different assigned executables.

5- Sample: A sample is a collection of simulation Realizations.

6- Realization: Realization is a collection of Parameters with different values which are generated according to specified probability distributions.

7- Parameter: The Parameter is a concept to manage the input values of an executable. The Parameter entity describes an input value. It provides additional information about the distribution used to generate the value and the required index to find the corresponding placeholder of the value in the input template file.

8- Template file: These files are text files with some additional terms for specifying the location of the probabilistic parameters in the body of the input files. The Model entity reads the content of the template file and replaces the placeholder with the parameter values of the current run of the simulation and generates new text files.

9- Result Converter: Result Converter provides the functions to retrieve, collect and store the values that are generated by a simulator. It uses Regular Expressions to recognize, extract and parse the values from the content of output files. In addition, it uses a list of Index components to select the values which are interesting for post-processing operations.

10- Index: Index is an item to keep the information about extracting and storing the data which are collected by the Result Converter.

11- Result Tree: The Result Converter stores the input and output values of simulations in binary files. The Result Tree is a hash table which manages the content of these files in the memory. The index of the Parameters is used as key to find the input and output values for whole realizations. The Result Tree component provides an interface between the binary files on the disc and the Result Converter components to load the content of the binary files, insert values into them and save them again as binary files on the disc.

Figure 23 illustrates how the described components are coupled together to build the Core component.

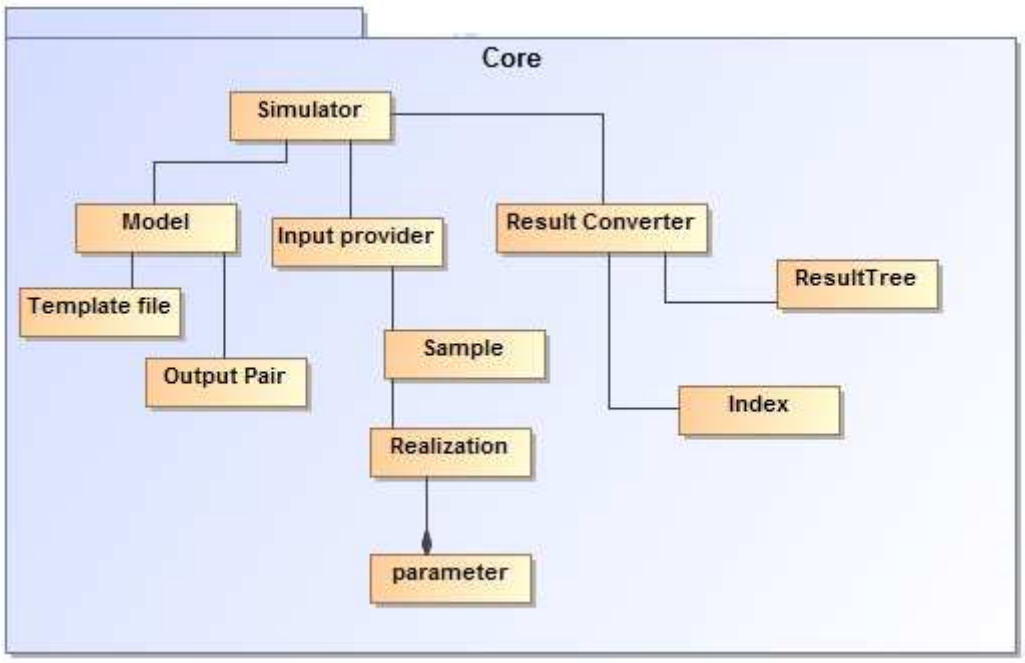


Figure 23. Simplified class diagram of Core

6.2.4 Analyzer Tool

The following entities are designed to implement the functionalities of the Analyzer Tool:

1- Chart Tool: This tool provides a graphical interface in order to configure the conversion of the simulation values into the charts. This program requires an XML file to start. The XML file contains a list of binary files that should be loaded in the Chart Tool .

   a. Panel: Panel provides a graphical user interface that allows the user to specify the configurations of the generating charts.

   b. ChartItem: ChartItem represents the specified XML nodes which address the binary files as input of the Chart Tool. This component will be parsed from XML file and created in the memory. It is the middle layer between Analyzer Tool and the binary file.

   c. Chart: Chart is a graphical item which can draw lines, plots or histograms.

   d. Chart Factory: Chart Factory uses the Chart Item component to extract the values from binary files and calculate them according to the configuration submitted by the user and finally creates the proper Chart objects. Furthermore, it provides the regression and additional quantile functions to represent the coherence between values.

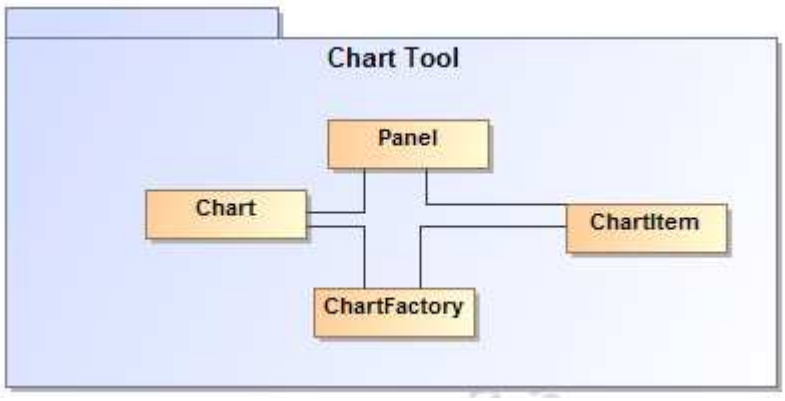


Figure 24. Simplified class diagram of the Chart Tool

2- Export tool: Export tool provides a graphical user interface to view the simulation results in the form of tables. In addition, this entity provides a functionality to calculate the correlation coefficient between input and output values of the simulations

   a. Panel: provides a graphical user interface that allows user to explore and specify the used functions to view the tabular form of the data

   b. Table: shows the selected simulation results in the form of the tables

   c. Export Item: this entity is an interface between the Export tool and the stored values in binary files. It reads the requested values from binary files and transforms them into the request format by Export Tool entity. In addition, this component provides a functionality to convert the values into the external file formats (CSV and XLS)

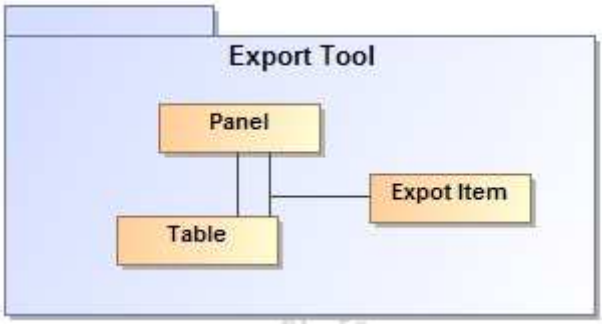


Figure 25. Class Diagram of the Export Tool

6.3 Development

The ReSUS platform consists of over 2,000 lines of C++ and Java code, using boost, Qt (Qt Company, 2016), QWT (Rathmann & Wilgen, 2016)and xlslib (xlslib, 2016) libraries. It contains five major components; GUI, Input Sampler, Core, Chart Tool, and Export Tool. We have separated these components in order to decrease the complexity of codes as well as improving the maintenance and extensibility. This also allows the other developers to use the ReSUS Platform components in their software projects.

The Graphical User Interface (GUI) is implemented in Java as a plugin for Eclipse (Clayberg & Rubel, 2008)with the help of the Graphical Editing Framework (GEF) (Rubel, Wren, & Clayberg, 2011) and of Eclipse Plugin Development (EPD). The wrappers of the HDF5 library make it possible to use its Application Programming Interface (API) in Java in order to develop the Input Sampler.

The Core component is developed as a C++ console project in Qt which utilizes the Boost and PCRE (Perl Compatible Regular Expressions) libraries for interpretation of the regular expressions. Furthermore, the Boost library is employed to parse the XML files and accomplish multithreading.

The components of the Analyzer Tool (Chart Tool and the Export Tool) are implemented in the C++ language using the Qt Widgets which are the primary elements for creating user interfaces in Qt applications. Using the results of the simulations, the QWT (Qt Widgets for Technical Applications) library draws the graphical charts while the xlslib library generates Microsoft Excel files (XLS format). The source code contains comments about the functionality of each code snippet.

Since each of the five aforementioned ReSUS components are implemented in a different language, the extensible markup language (XML) files are used as a common media among them. The XML files contain the configurations and structure of simulations. These files are generated by GUI and processed by Core, Chart Tool, Export Tool and Input Sampler.

The Core, Chart Tool and Export Tool are implemented in the C++ language by Qt Creator as separate projects. The 32-bit version of MSVC2012 from Microsoft Developer Toolkit is assigned to the Qt Creator to compile these projects. The GUI and the Input Sampler are implemented in Java language by Eclipse IDE and are compiled with Java Standard Edition (SE) Development Kit (JDK) 7.

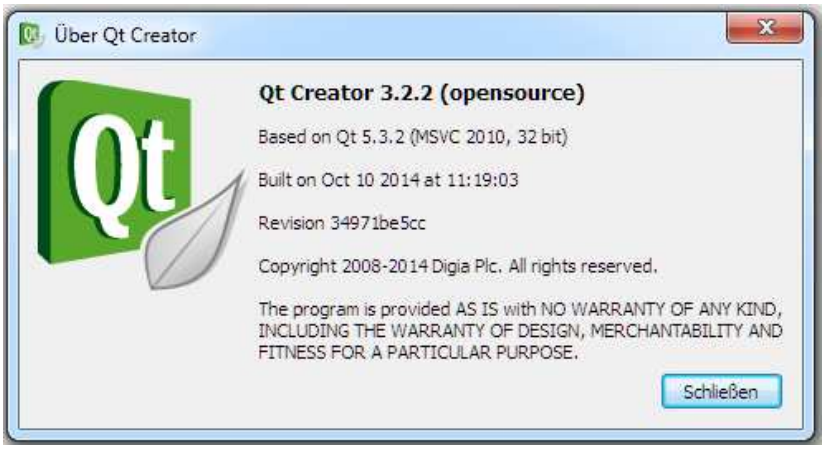


Figure 26. Qt Creator used in order to develop the ReSUS Core and Analyzer Tool

6.4 Test

The unit tests are generated as subproject for each of the Core, Charts and export library projects. All the core project classes are tested under the Boost Test Suite (Rozental, 2016) to validate the member functions of the classes.

The majority of Graphical User Interface project is about reading and writing the data and is not considered in the automated testing process. These cases are tested manually during the development phase. JUint Test Framework (JUnit, 2016) is utilized to implement the test cases for sample generator functions.

Moreover, a user test is performed and protocolled with regard to the usage points included in the user handbook. More details about test cases are available in the developer handbook (Ghofrani J. , 2016) as well as source code packages of the project.

6.5 Deployment

The ReSUS installation package is delivered as a portable software package which includes the executables of the Core, Chart Tool, Export Tool. The required static and shared libraries (e.g. HDF5,

PCRE, QWT, and libraries required for MSVC2012) which are necessary for running the components of the ReSUS Platform are also involved in their corresponding folders. Moreover, the plugin developed for Eclipse IDE is compiled and installed on Eclipse and attached to the installation package. Furthermore, the files required for executing the examples presented in Section 7 and in the user handbook are also available in the installation package of ReSUS.

ReSUS is compatible with Windows Vista, Windows 7 (either 32-bit or 64-bit versions), Windows 8, and Windows 8.1. However, this platform is not compatible with Windows Server 2003, Windows XP, Windows RT and ARM-based CPUs. It requires a minimum of 1 GB disk space and 1 GB of RAM Along with 16-bit of color depth (32-bit recommended).

7 Evaluation

In this chapter, we evaluate the ReSUS Platform through three types of simulation examples. The first part of the chapter focuses on the simulations, which are already available in other tools such as MatLab, the starter version of the ReSUS (Li, 2015) and Ecolego (Facilia, 2016). The second part of this chapter shows the ability of the ReSUS Platform to utilize the external simulation programs in order to perform probabilistic simulations. The additional functionality of the ReSUS Platform to make complex models and perform parallel simulations is illustrated by using the RockFlow (Kolditz, Kaiser, Habbar, Rother, & Thorenz, 2003) and Level-E (Nuclear Energy Agency, 1989) examples in the third part.

The details of these simulations are also available in the user handbook (Ghofrani & Li, 2016) for the ReSUS Platform. This handbook explains the steps required to prepare these simulations, run them and analyze their results. The simulations implemented in ReSUS are also available in the examples folder of the ReSUS Platform installation package.

Furthermore, the XML files which describes the probabilistic simulation workflows and their input Template files can be found in the appendix of this this thesis.

7.1 Evaluation through existing approaches

In this part, we present the ability of the ReSUS platform to implement existing examples of uncertainty and sensitivity analysis cases in MatLab, Ecolego and the starter version of ReSUS.

7.1.1 Ishigami

The Ishigami (Ishigami & Homma, 1990) function is a widely used example in uncertainty and sensitivity analysis. The equation below shows this function and its input parameters as x1, x2 and x3:

$$f(\mathbf{x}) = \sin(x_1) + a\sin^2(x_2) + b\,x_3^4\sin(x_1)$$

The input parameters are random values uniformly distributed on the interval [-π,π] with. The values of “a” and “b”, used by Crestaux et al. (Crestaux, Martinez, Le Maitre, & Lafitte, 2007) and Marrel et al. (Marrel, Iooss, Laurent, & Roustant, 2009), are: a = 7 and b = 0.1.

Accounting for these details, the following C++ code was written. It reads the x1, x2 and x3 from a file named input.txt and stores the f(x) as output file with the name output.txt.

```
int main(int argc,char *argv[]) {
        if(argc<2) {
                printf("\nERROR READING PARAMETRES FILE. \n
                please use input file as argument .\n
                example : ishigami.exe input.txt\n");
                return -1;
        }
        FILE* file = fopen("input.txt","r");
        printf("\nOpening input file ...");
        double a,b,c;
        fscanf(file,"%lf\t%lf\t%lf",&a,&b,&c);
        printf("\nReading values... ");
        fclose(file);

        printf("\ncalculating ...");
        double result=sin(a)+7*sin(b)*sin(b)+0.1*(c*c*c*c)*sin(a);

        printf("\nSaving results");
        file = fopen("output.txt","w");
        fprintf(file,"%lf",result);
        fclose(file);
        printf("\nEndig program\n");
        return 0;
}
```

We compiled this code and assigned the executable generated to the ReSUS platform. Next, we generated 3000 random values between -3.1415 and +3.1415 for each input parameter (X1, X2, X3) and a dummy parameter (X4).

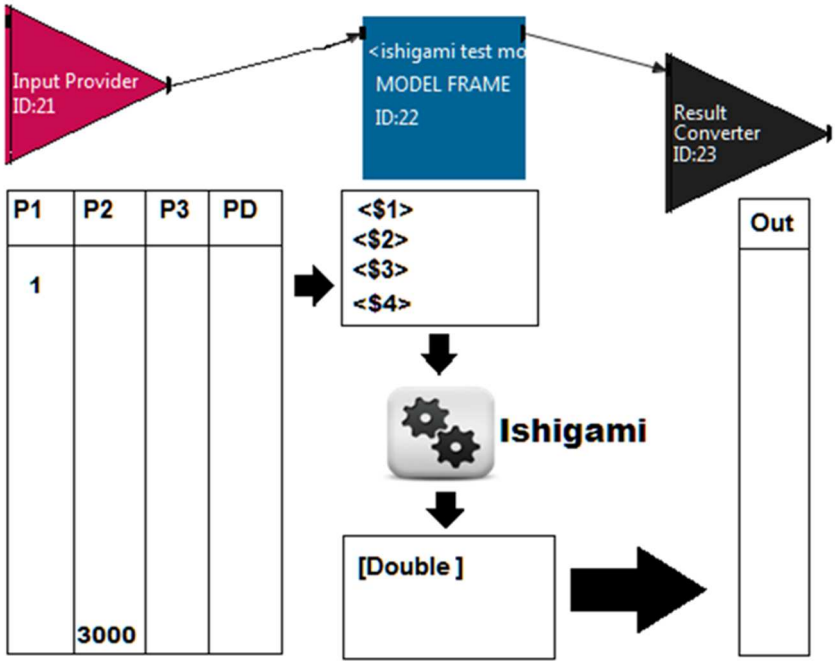


Figure 27. The process of Probabilistic calculations of the Ishigami Model by the ReSUS Platform

Figure 27 illustrates the general process of running the Ishigami model with the provided probabilistic input values, replacing the random values with input parameters, and extracting the results from the output file.

The scatterplots generated with the Chart Toolbar is shown in Figure 28 below. In addition to the three input variables; we have generated one dummy parameter additionally to test the correctness of the calculations; the dummy should be marked “insensitive” in any sensitivity analysis. The values of the dummy parameter are plotted against the output values, and the results for all four input parameters are illustrated in Figure 28

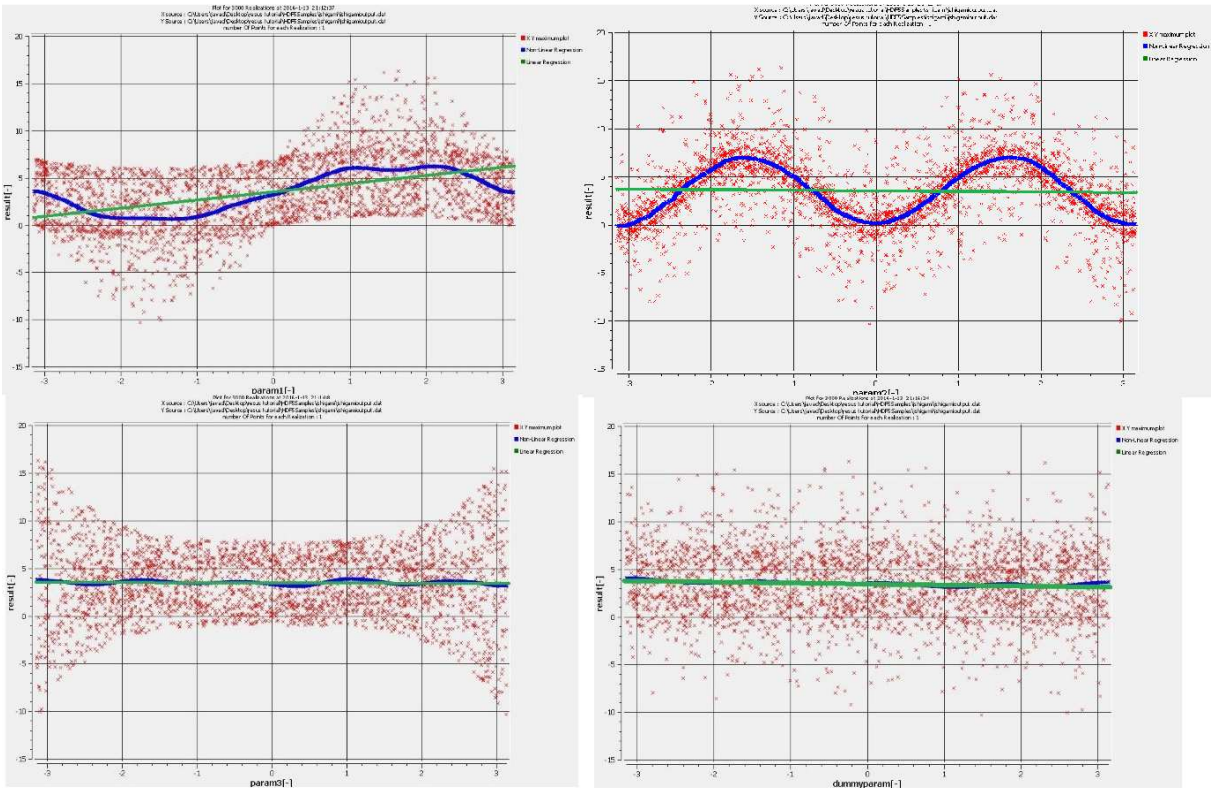


Figure 28. Scatterplots generated by ReSUS Platform according to the results of Ishigami model

In order to validate the sensitivity analysis result produced by the ReSUS Platform the following charts (Figure 29) illustrate the result of executing the MatLab code with 3000 realizations:

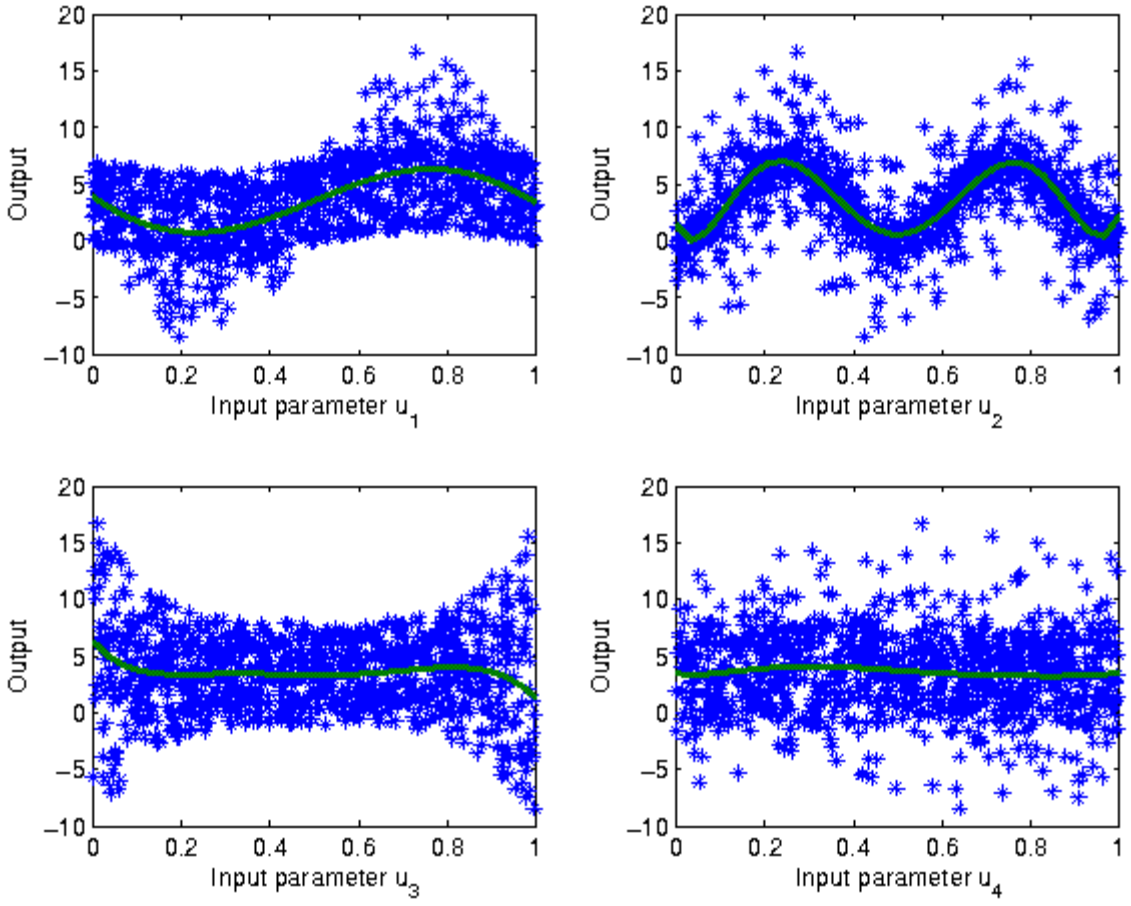


Figure 29. Ishigami simulation results in MatLab (Plischke E. , 2016)

It can be seen from the comparison of the graphics that ReSUS Platform (figure above) produced results similar to the ones from MatLab shown in Figure 29.

7.1.2 Level-E

Our second example shows the ability of the ReSUS Platform of simulating and analyzing a simple geological repository model which is already implanted in the preceding version of ReSUS (Li, 2015). This model is implemented as a deterministic simulation which simulates the transport of the radioactive particles in a two-layered system. Some of the important radionuclides (I-129, Np-237, U233 and Th-229) are released from the source term to the geosphere layers. The nuclides are transported through the layers and reach the biosphere where the radionuclide concentration will be calculated to dose values by a simple biosphere model equation. The release rate from the source term, the transport properties of the geosphere layers and the some of the parameters of the biosphere are considered as probabilistic parameters of this model (see Table 2).

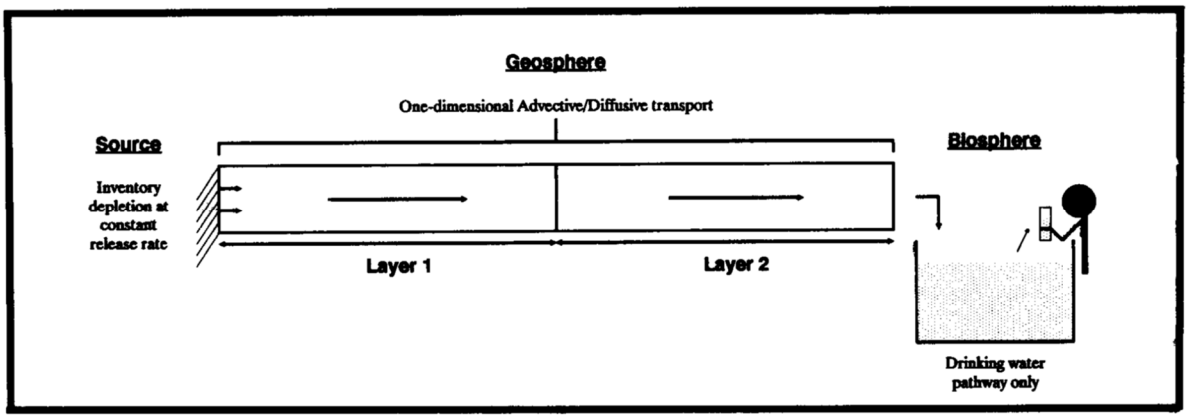


Figure 30. The implemetented model with Level-E for a simple repository system (Nuclear Energy Agency, 1989)

The gtmle.exe is the Level-E executable which reads the simulation parameters and values from the file named input.sam and generates three output files with .DAT extension: geosph1.dat, geosph2.dat and output.dat.

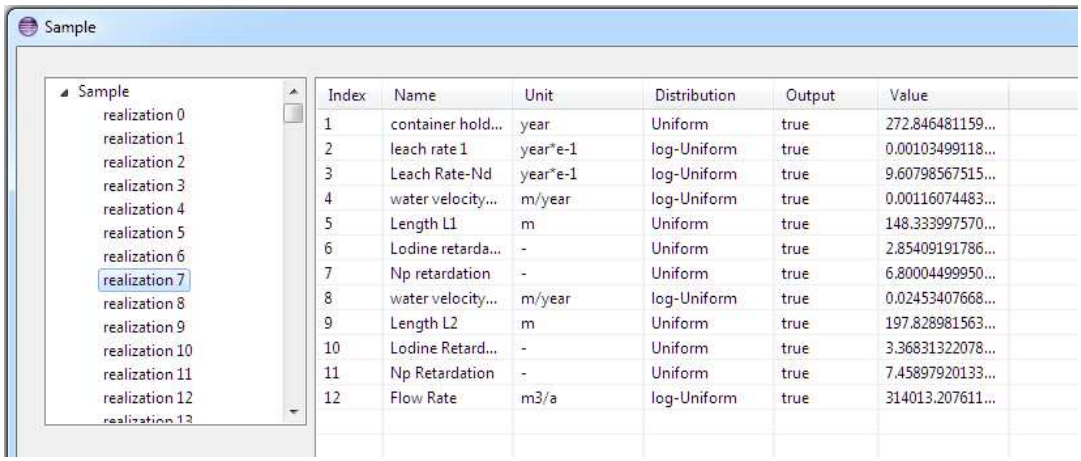
Compared to the Ishigami example presented in the previous part, the Level-E program is just an executable, which is assigned to the ReSUS Platform. The input.samfile is generated according to the following template:

```
0
1
12
0
<$1>  <$2>  <$3>  <$4>  <$5>  <$6>  <$7>  <$8>  <$9>  <$10>  <$11>  <$12>
```

The parameter configurations listed in Table 2 are used to generate 3000 realizations (depicted in Figure 31 and Figure 29) and to perform a probabilistic simulation on the designed Level-E model.

Table 2. Paramter Configurations for the probablistic simulation of Level-E Model. After (Li, 2015)

Index	Parameter	Unit	Distribution	Interval	Description
1	*Container Hold time*	year	Uniform	100-1000	Period of time up to container failure
2	*Leach rate I*	l/a	Log-Uniform	$0{,}001\text{-}10^{-2}$	Release rate of iodine
3	*Leach rate-Nd*	l/a	Log-Uniform	$10^{-6}\text{-}10^{-5}$	Release rate of Np-chain
4	*Water Velocity- L1*	m/a	Log-Uniform	0,001-0,1	Distance flow rate in Geo-layer 1
5	*Length-L1*	m	Uniform	100-500	Length of Geo-layer 1
6	*Iodine-Retardation*	-	Uniform	1-5	Retardation factor for iodine in Geo-layer1
7	*Np-Retardation*	-	Uniform	3-30	Retardation factor for Np-chain in Geo-layer 1
8	*Water welocity-L2*	m/a	Log-Uniform	0,01-0,1	Distance flow rate in Geo-Layer 2
9	*Length-L2*	m	Uniform	50-200	Length of Geo-Layer 2
10	*Iodine-Retardation2*	-	Uniform	1-5	Retardation factor for iodine in Geo-Layer 2
11	*Np-Retardation*	-	Uniform	3-30	General conversion factor for the retardation factor of Np-chain in Geo-Layer 2
12	*Flow rate*	m^3/a	Log-Uniform	$10^5\text{-}10^7$	Looked volume of river water in the biosphere per year



Index	Name	Unit	Distribution	Output	Value
1	container hold...	year	Uniform	true	272.846481159...
2	leach rate 1	year*e-1	log-Uniform	true	0.00103499118...
3	Leach Rate-Nd	year*e-1	log-Uniform	true	9.60798567515...
4	water velocity...	m/year	log-Uniform	true	0.00116074483...
5	Length L1	m	Uniform	true	148.333997570...
6	Lodine retarda...	-	Uniform	true	2.85409191786...
7	Np retardation	-	Uniform	true	6.80004499950...
8	water velocity...	m/year	log-Uniform	true	0.02453407668...
9	Length L2	m	Uniform	true	197.828981563...
10	Lodine Retard...	-	Uniform	true	3.36831322078...
11	Np Retardation	-	Uniform	true	7.45897920133...
12	Flow Rate	m3/a	log-Uniform	true	314013.207611...

Figure 31. A realization for the Level-E model

The Chart Tool of the ReSUS Platform is utilized to visualize the input and output values of the simulation. On one hand, Figure 32 illustrates the time evaluation of time dependent output values which are extracted from geoshp2.dat and output.dat are drawn in the form of line charts.

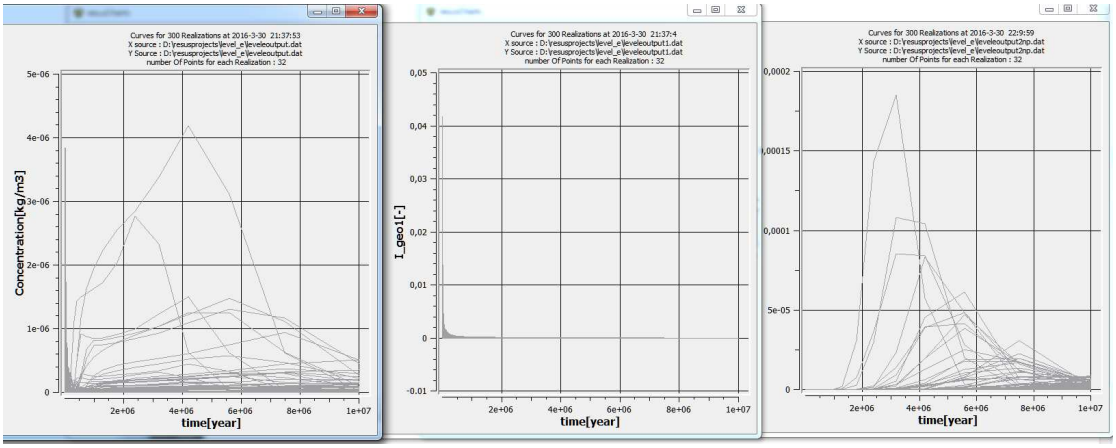


Figure 32. Output line charts of Level-E simulation by ReSUS

On the other hand, Figure 33 shows the time evolution of time dependent results of the Level-E simulator which are generated by the preceding version of ReSUS (Li, 2015).

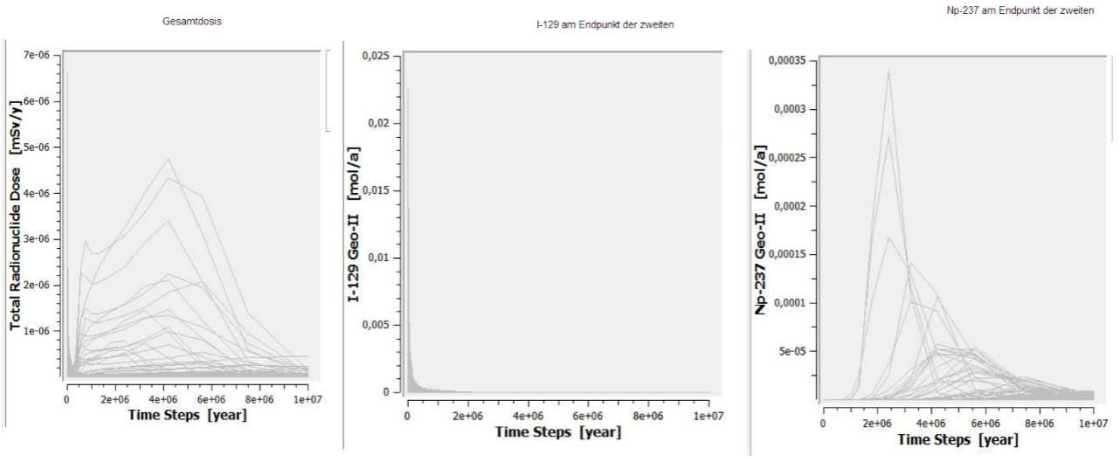


Figure 33. The line charts of the ReSUS Starter version for three different concentration values

Moreover, the histogram generated for the concentration values in output.dat file and the geosph2.dat file from the ReSUS Platform Analyzer Tool are depicted in Figure 34 and Figure 35which are similar to the histograms of the concentration values in the preceding version of ReSUS (shown in Figure 35).

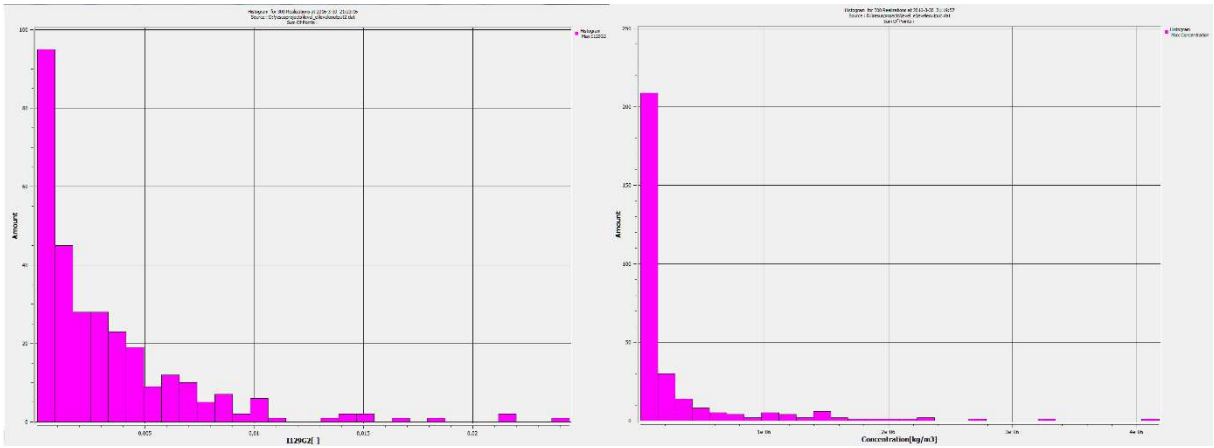


Figure 34. Displaying values for concentraion paramter (output of Level-E simulation) as histogram for two output files: output.dat and geosph1.dat

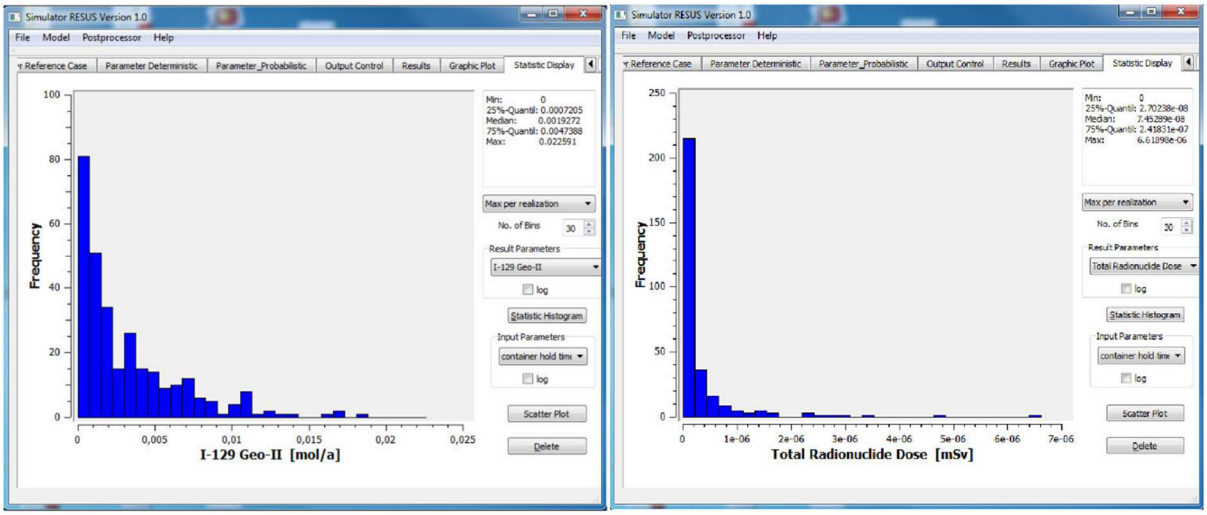


Figure 35. Generated histograms by preceding version of ReSUS for for concentration values (output.dat and geosph1.dat) of Level-E simulation. (Li, 2015)

Finally, the plotted values of the input parameters (Water velocity L1 and Water velocity L2) against the output parameters (concentration) are generated by Chart Tool (demonstrated in Figure 36 and Figure 38) and are compared with the scatterplots of the preceding version of ReSUS (illustrated in Figure 37 and Figure 39).

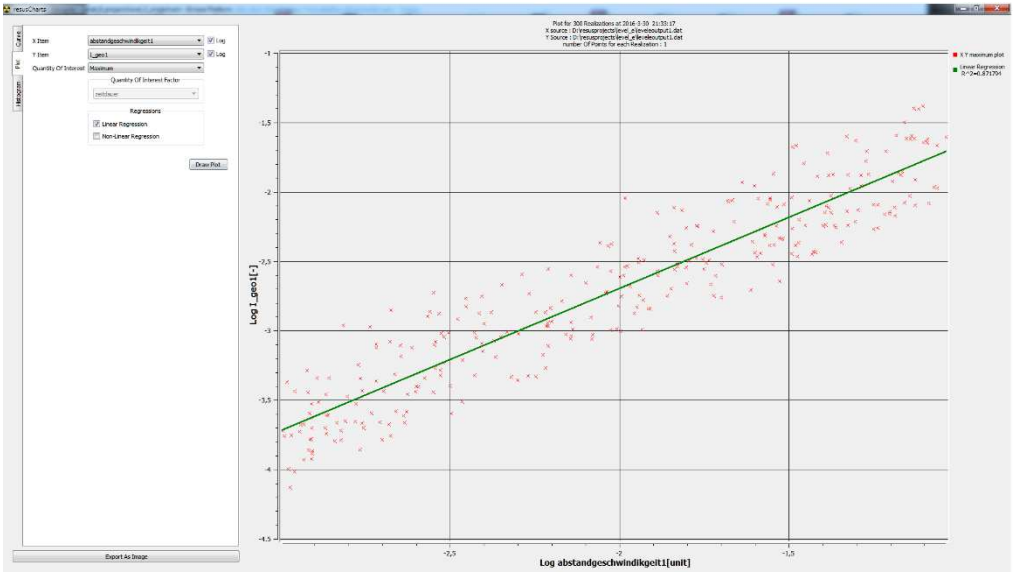


Figure 36. Scatterplot of an input parameter (x axis: Water velocity L1) against an output (concentration of Iod129 in Geolayer1) extracted from the Level-E simulation by ReSUS

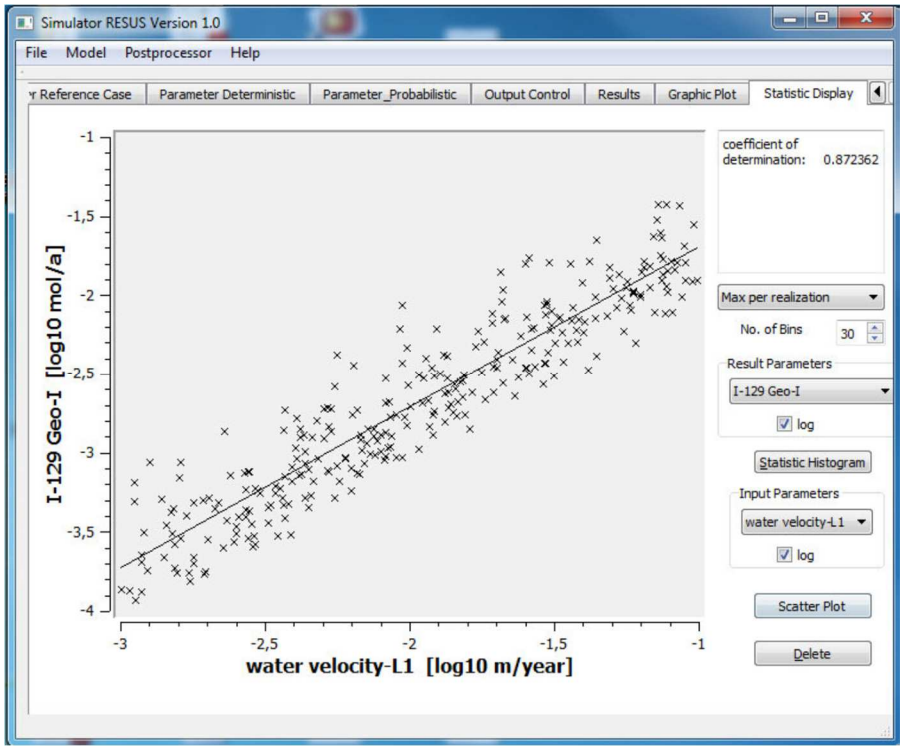


Figure 37. Scatterplot of an input parameter (x axis: Water velocity L1) against an output (concentration of Iod129 in Geolayer1) of Level-E simulation by the preceding version of ReSUS (Li, 2015)

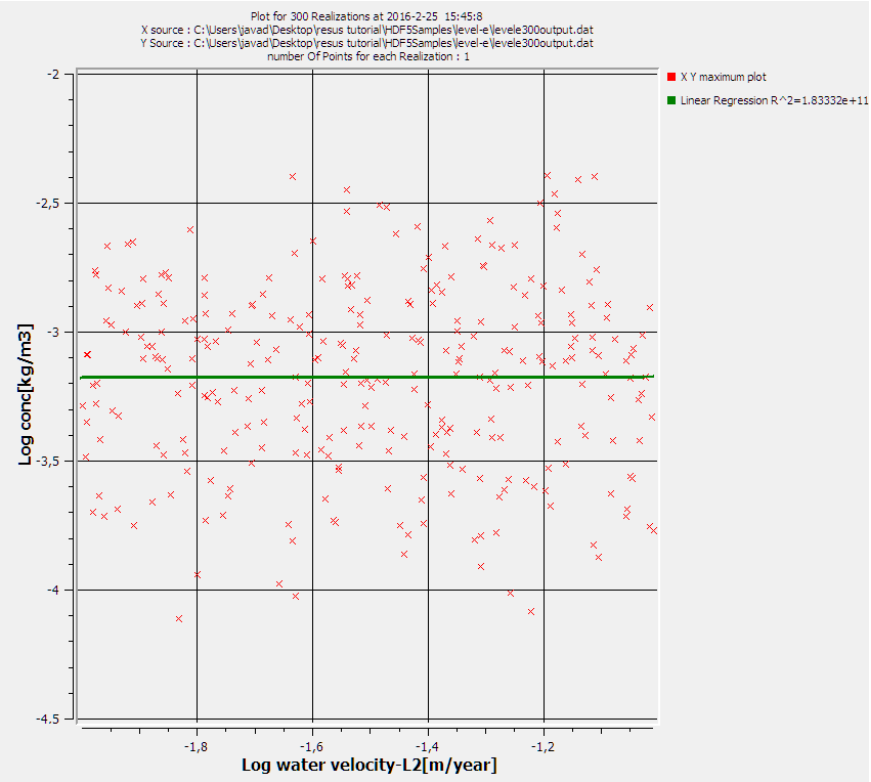


Figure 38. Scatterplot of an input parameter (X axis: Water velocity L2) and an output (Y axis: Concentration) values of the Level-E simulation by ReSUS

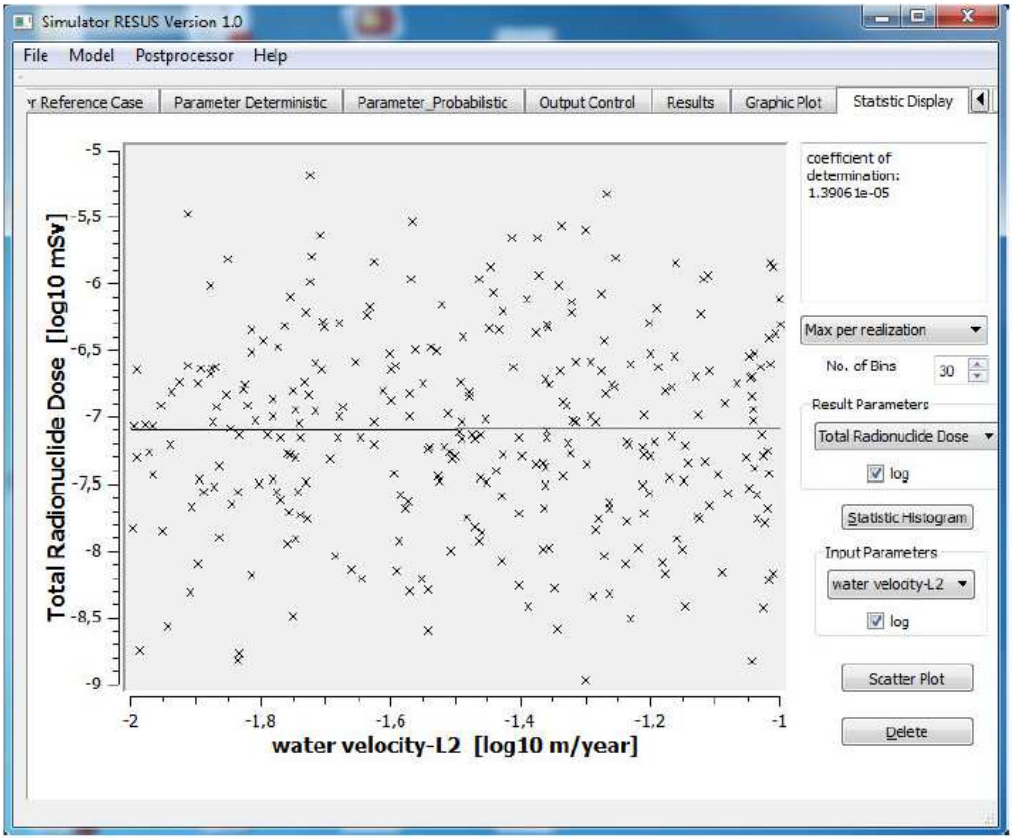


Figure 39. Scatterplot of an input parameter (X axis: Water velocity L2) and an output (Y axis: Concentration) values of the Level-E simulation by the preceding version of ReSUS (Li, 2015)

In addition to the presented plot and histogram charts above which are generated using Maximum as Quantity of Interest, Figure 40 illustrates a plot chart which is generated by “Time of Maximum Value” as Quantity of Interest.

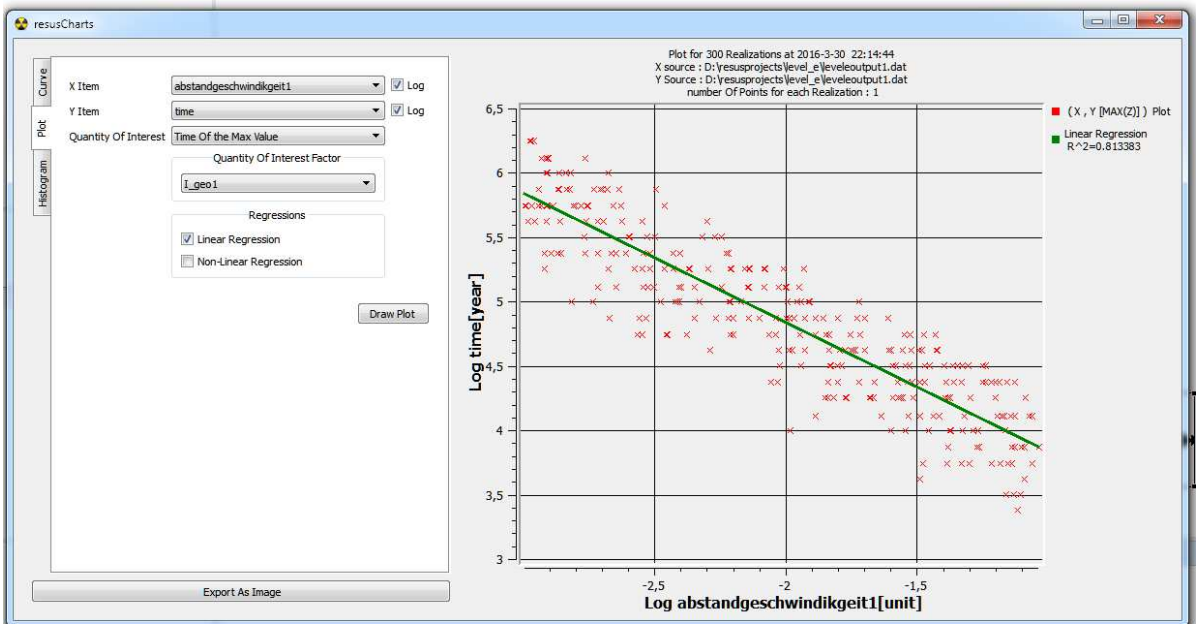


Figure 40. Time of Maximum for Iodine 129 from layer 1 against Water velocity (X axis) simulated by Level-E assigned to ReSUS

This demonstrated example of Level-E simulation shows the ability of ReSUS to perform the simulations that are already embedded in its preceding version. In contrast to the working mechanism of the preceding version, the external executable does not require any programming knowledge and code manipulation in the ReSUS platform.

7.1.3 Ecolego

As the third example, the ReSUS platform is compared with Ecolego (Facilia, 2016) by simulating a mathematical radioecological model for determining the radiation dose exposition. The analyzed results from both platforms are used to benchmark the ReSUS Platform.

The simulation model studied in this example is developed regarding the German approach in the project BioMoSA (Pröhl, et al., 2005). This mathematical model with 94 input parameters, in which 32 of them are probabilistic, is implemented in the Ecolego software as well as a C++ program (Ciecior & Ghofrani, 2016).

The compiled executable of the C++ project is assigned to the ReSUS Platform and the probabilistic simulation with 32 input parameters of this model is performed.

The Figure 41 depicts the generated Values for Dose Conversion Coefficient from both software platforms as empirical cumulative distribution functions.

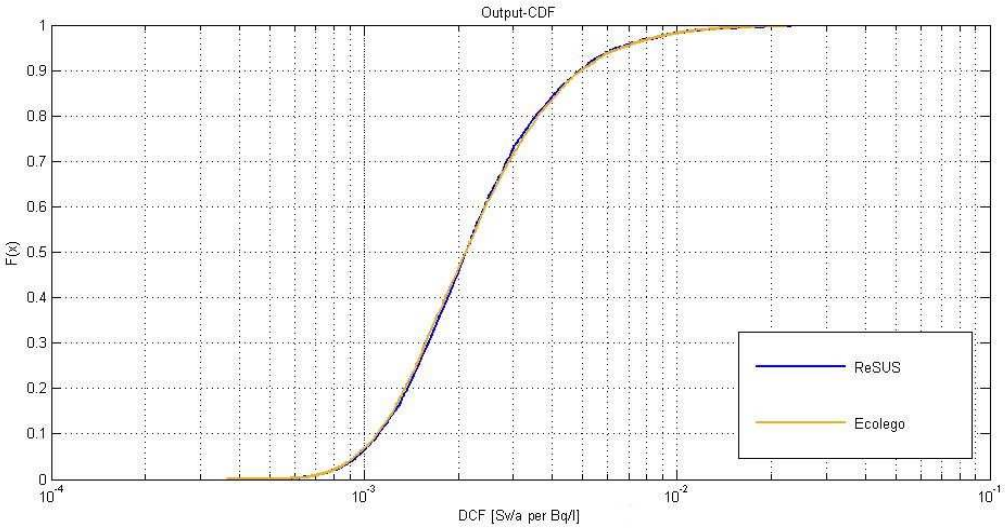


Figure 41. Comparison of ReSUS and Ecolego usingCDFs

Figure 42shows the values of the comparison of the Pearson's and Spearman Correlation Coefficients, calculated by the Export Tool of The ReSUS Platform and the Ecolego software.

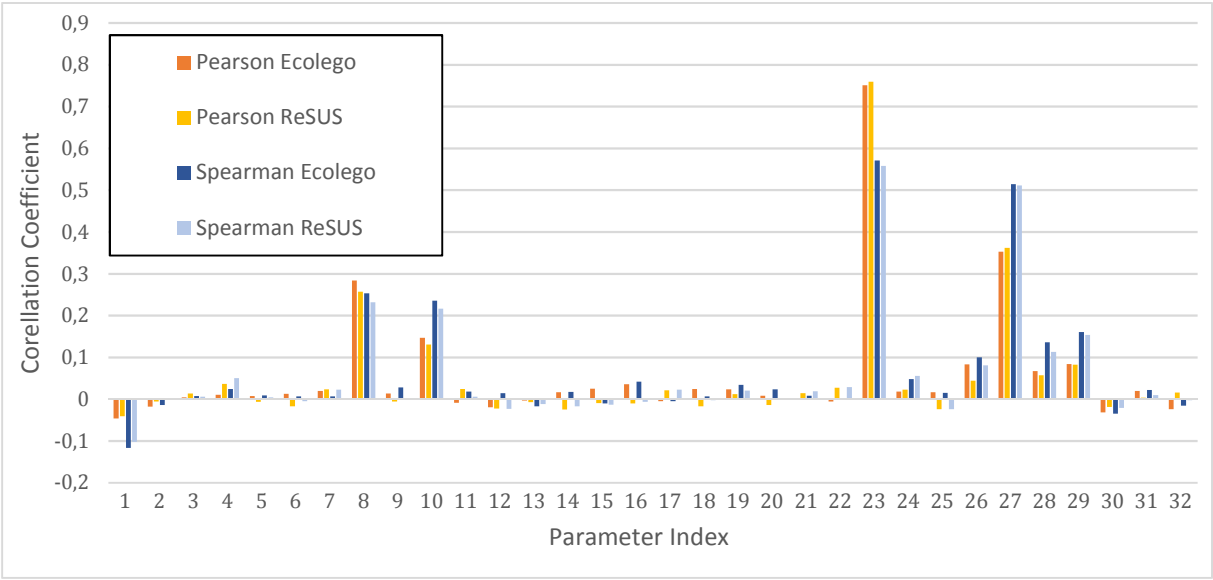


Figure 42. Comparison of Spearman and Pearson's correlation coefficients for parameters of the biosphere model and the dose conversion factors, simulation results of ReSUS and Ecolego

7.2 Implementing probabilistic simulations with the help of the ReSUS Platform

The objective of this part is to present the ability of ReSUS to utilize the deterministic simulation tool in a probabilistic way. Furthermore, it is shown to which external deterministic simulators ReSUS is compatible. The examples presented in this part are not benchmarked by any similar implementation in other software tools and are just demonstration examples.

7.2.1 PHAST

PHAST (Parkhurst, Kipp, Engesgaard, & Charlton, 2004) is a computer program that simulates the reactive transport of different chemical species by fluid phases in ground-water flow systems.

We have implemented a PHAST model and assigned it to the ReSUS platform in order to probabilistically analyze the radioactive nuclide transport process in the overburden of a disposal system. This model is just a test to demonstrate the ability of the ReSUS Platform to work with the PHAST program and to perform the probabilistic simulations as well as uncertainty and sensitivity analysis on the delivered results. Due to the fact that the PHAST model is still under construction, the depicted results are just to show the ability of ReSUS and the values are not yet validated for realistic usage.

The example studied here addresses the migration of a contaminant (Uranium) through an aquifer. The aquifer is the overburden above a disposal host rock. The overburden consists of three layers. The middle one is a clay layer with lower permeability. However, the layers above and below the clay are sand stone with higher permeability. It is supposed that the Uranium is released from the host rock and contaminates the ground water in the aquifer. The released Uranium concentration is given as release source into this model and works as boundary condition. It is considered as the only probabilistic parameter. The samples for this parameter are generated with the log-Uniform distribution between max_value = 10.0 and min_value = 1.0E-4.

The calculated time dependent values for U (calculated Uranium concentration at a certain point of the model) are shown in the figure below:

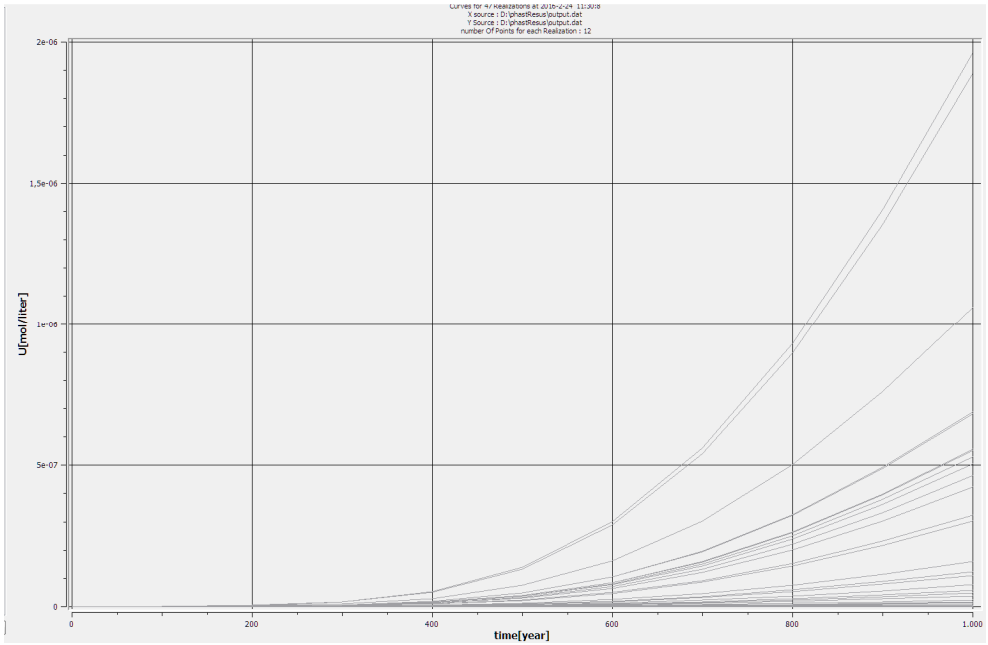


Figure 43. Time dependent values of U (uran Concentration) generated by the PHAST Model and drawn by the ReSUS Chart Tool (different realizations)

7.2.2 TOUGH2

TOUGH (Transport of Unsaturated Groundwater and Heat) (Pruess, 1991) simulates, amongst other things, deterministically the transport of a radionuclide in a disposal system. There were two challenging points when assigning TOUGH to the ReSUS Platform, which are described below:

1- The TOUGH2 reads its simulation parameters directly from the standard console. This method of reading the input values makes it hard to use the template based methods for providing input parameters

2- Despite our manipulations of TOUGH2, it cannot generate the output values in the column format. The time dependent output values of TOUGH2 are printed in the form of a matrix block of values for each time step, instead of columns (Figure 44)

```
1  TIME =  1.00000000E+00
2   3.81612664E-14  3.81612664E-14  1.14357247E-07  0.00000000E+00  1.14357247E-07
3   1.45604576E-27  1.45604576E-27  4.36330869E-21  0.00000000E+00  4.36330869E-21
4   5.55555269E-41  5.55555269E-41  1.66482345E-34  0.00000000E+00  1.66482345E-34
5   2.11972497E-54  2.11972497E-54  6.35214540E-48  0.00000000E+00  6.35214540E-48
```

Figure 44. The Output format of the TOUGH2 and the time dependent concentration values, which must be retrieved

To overcome these problems, our suggestion is to use batch files which call TOUGH2 and redirect the content of the input file (which is generated by ReSUS using the ASCII template file) to TOUGH2 with the help of the Input redirection character "<". The generated batch file is assigned to the ReSUS Platform.

In order to extract the time and concentration values generated by TOUGH2 from its output file, we wrote a simple program, which recognizes the TIME word in a text file (output of TOUGH2) and prints the value using a certain index (e.g. 6^{th}). This additional program is an intermediate step between the TOUGH output and its assigned Result Converter. The Result Converter converts the generated values from the additional program and stores them.

The configurations listed in Figure 45are used to generate 10realizations and to perform the probabilistic simulation by ReSUS.

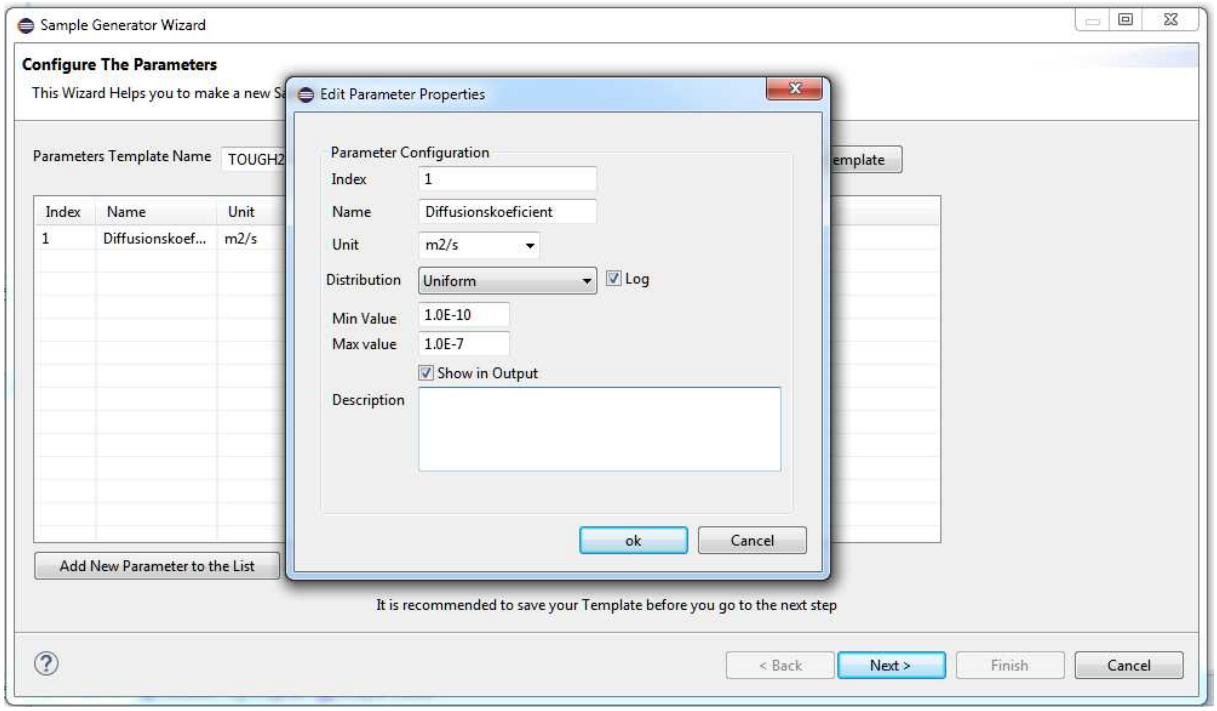


Figure 45. Parameter configuration of the TOUGH2 model

The TOUGH2 model investigated in this example is a radionuclide transport model, which describes the transport process in a disposal system in clay formation. The model is constructed taking into

account the clay host rock, boreholes, tunnels and the shaft system. It is considered that the high level waste canisters are emplaced into the boreholes and the disposal system is backfilled. After the failure of the canisters, e.g. due to corrosion, the radioactive nuclide will be released from the disposal borehole. It is supposed that the diffusive and advective transport will take place in this model. An observation point is set at the surface of the clay host rock.

We have used the TOUGH2-EOS9nT (Moridis, Wu, & Pruess, 1999) module as our external program to perform this consequence analysis. However, the ground water flow data between the finite elements in this model is calculated by FLAC-TOUGH program. We have modified the source code of TOUGH2-EOS9nT to calculate the transport process by using the external flow data from the FLAC-TOUGH program in each calculation iteration. The absorption coefficient Kd and the diffusion coefficient D of nuclides will be considered in this model as probabilistic parameters (Bänecke, et al., 2015).

The generated values for time and concentration are illustrated in the form of the line chart and the scatterplots (used Maximum as the function for Quantity of Interest) below:

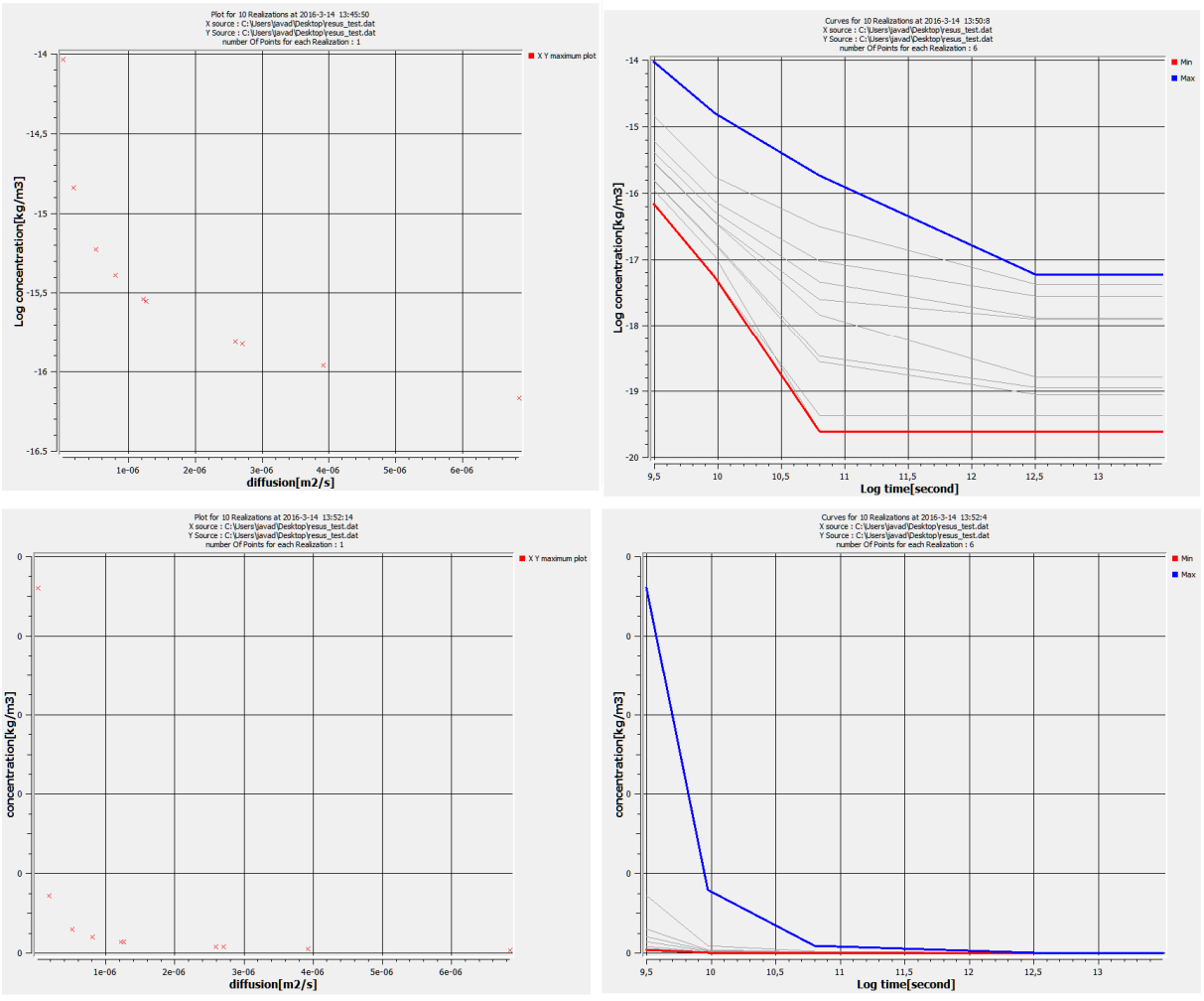


Figure 46. Results of the probabilistic simulation of TOUGH2

The results illustrated in Figure 46 show the results parameter evolution over the time from the first 10 realizations of the uncertainty and sensitivity analysis from this model using Maximum as Quantity of Interest. Due to the long calculations times the simulations are not yet completed.

7.3 Additional functionalities of the ReSUS Platform

In both previous parts of this chapter, the abilities of the ReSUS platform to perform probabilistic simulations using external deterministic models are presented. In this part, we present a powerful functionality of the ReSUS Platform for simulating the models containing more than one simulating component. In addition, we show the functionality and flexibility of Chart Tool and Export Tool in such multi-component simulations.

7.3.1 RockFlow

RockFlow (Kolditz, Kaiser, Habbar, Rother, & Thorenz, 2003) is a numerical simulation tool, which is based on the finite elements method. It is developed in order to simulate groundwater flow and contaminant migration processes.

This part of our work aims at showing the flexibility of the ReSUS Platform in design and execution of simulation workflows with more than one simulator component. The model presented in this subsection consists of two components: a granite fracture-matrix system and a transport simulation in the overburden above the granite host rock. Radionuclides, which might migrate from a repository for nuclear waste in crystalline rock, are transported by the water in a fracture network. The transport in a one-dimensional fracture is represented in the first model component. The second model component represents the groundwater flow in a porous medium above the host rock, which is modeled as a two-dimensional RockFlow transport model. The nuclide concentration leaving the fracture will be given as point source term at the bottom of the left side of the overburden model, and it will migrate from left to the right side. Afterwards the concentration of the nuclide particles is measured at the right side of the overburden model (second component). Figure 47 illustrates the designed simulation workflow by ReSUS according to the structure of this model. In this design, one can see that the first three simulation components represent the first layer of the model, and the second Model layer is mapped by the other three simulation components. The Input Provider component in the middle takes the responsibility of guiding the simulation flow from first model workflow to the second one. The Model Frame components manage the execution of the RockFlow simulator for each layer. The corresponding specifications and probabilistic parameters of each layer are described in the provided input template for the Model Frame components.

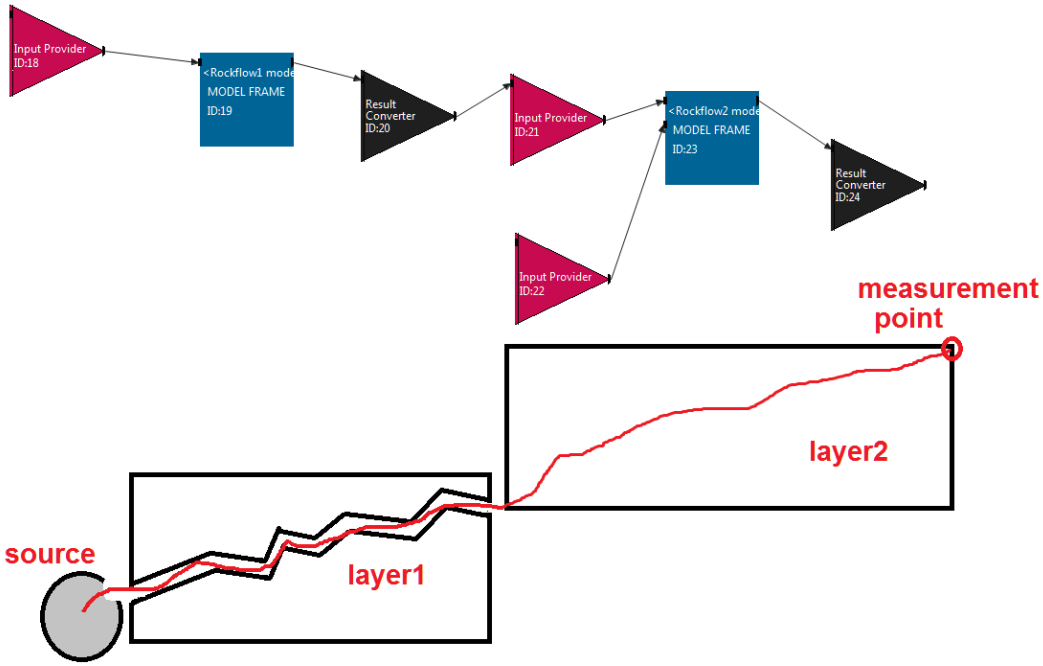


Figure 47. Two layered model simulation

The ReSUS Platform is able to transfer the values retrieved from the output of one model to the input provider of next model in the simulation workflow. This feature of the ReSUS Platform makes it a good candidate to simulate complicated models.

In this simulation case, the ReSUS Platform automatically transfers the calculated time dependent concentration values from the first Model Frame into the second Model Frame and registers them as boundary conditions in the input file of the second RockFlow executable.

The following configurations are used to generate the parameters, which are considered as probabilistic in the first RockFlow model:

Table 3. Required configuration for generating probabilistic simulation values for first RockFlow simulator

Index	Name	Unit	Distribution	Min	Max	Output	Description
1	Porosity_fracutre	-	Uniform	0.8	1.0	True	
2	Permeability	m2	Log-Uniform	1.0E-10	1.0E-8	Ture	
3	Porosity	-	Log-Uniform	0.2	0.5	True	
4	Diffusion	m2/s	Uniform	1.0E-12	1.0E-10	True	
5	Kd	kg/kg	Uniform	5.0E-5	5.0E-4	true	

The indices of the probabilistic parameter in the second model, presented in Table 4, start with number 2 because the index number 1 is already reserved for the delivered time and concentration values as line charts from results of the previous model.

Table 4. Parameter configurations for the second RockFlow model

Index	Name	Unit	Distribution	Min	Max	Output	Description
2	Kd	kg/kg	Uniform	5.0E-5	5.0E-4	True	
3	Porosity	-	Uniform	0.4	0.05	Ture	
4	Permeability	m2	Log-Uniform	1.0E-13	1.0E-11	True	
5	Dispersions length	m	Uniform	5	10	True	

Figure 48 illustrates the time dependent concentration values retrieved from first model while Figure 49 shows the time dependent concentration values calculated for the end point of the second model. The time shift caused by the transport of the radionuclides from the first layer to the second layer is visible in the lines.

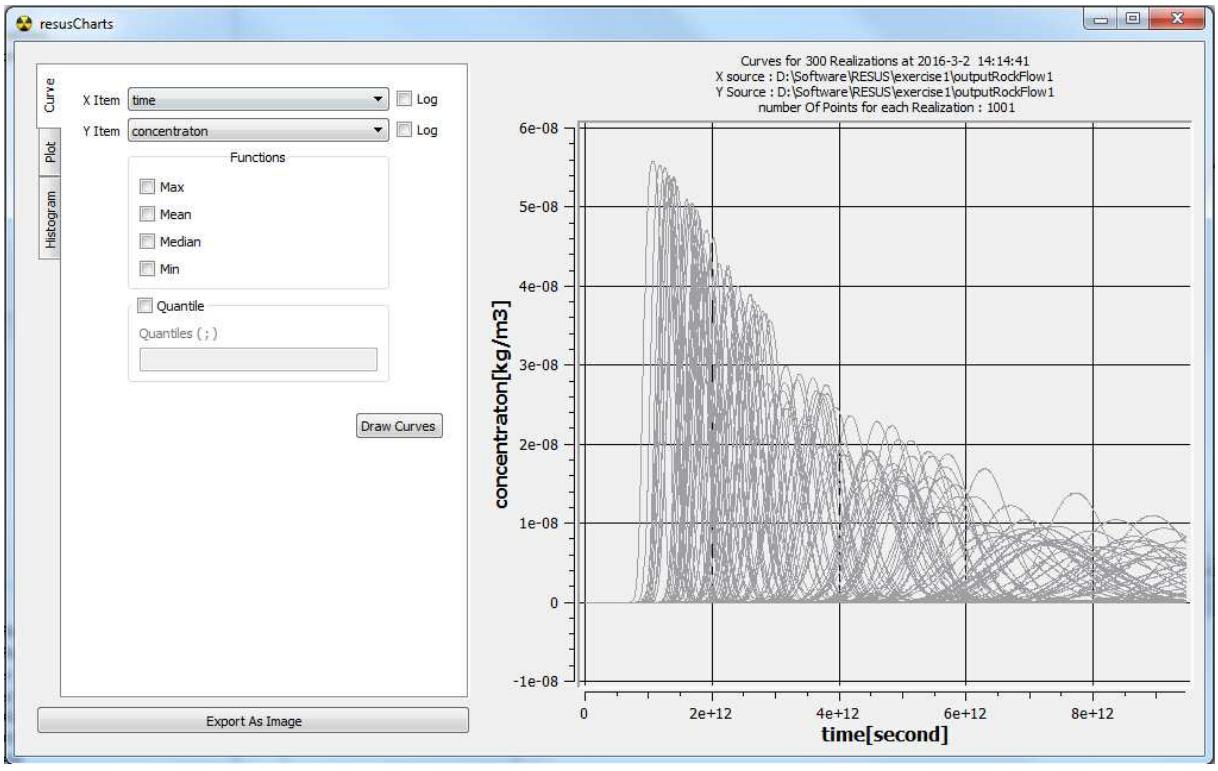


Figure 48. Line charts showing the output values of the firsf rockflow model

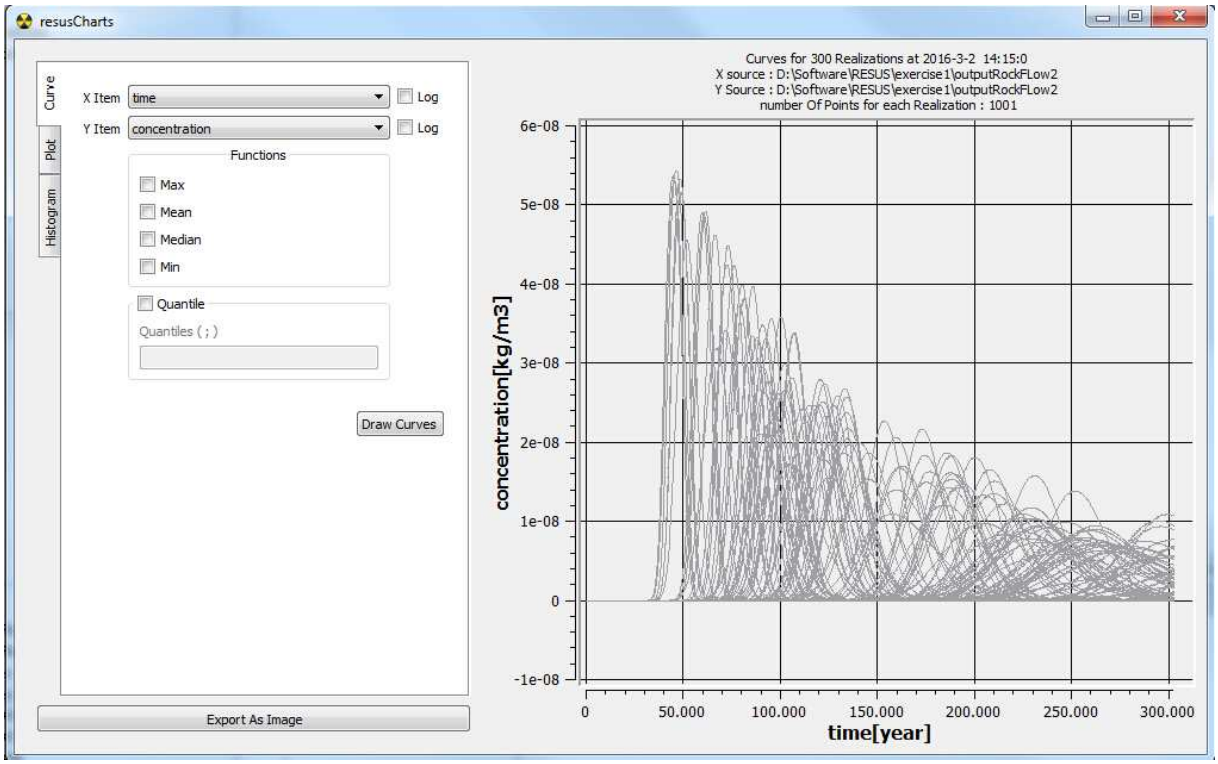


Figure 49. Line charts showing the output values of the second rockflow model

ReSUS can also perform sensitivity analysis on the input and output values of multi-layered models. In the Plot tab of the Chart Tool, the input and output parameter values from the first or the second layer of model can be selected to be plotted. The scatterplot shown in Figure 50 is an example of a sensitivity analysis using the ReSUS platform. It demonstrates the relationship between the permeability parameter from first model layer with the Concentration values from the output of the second model part.

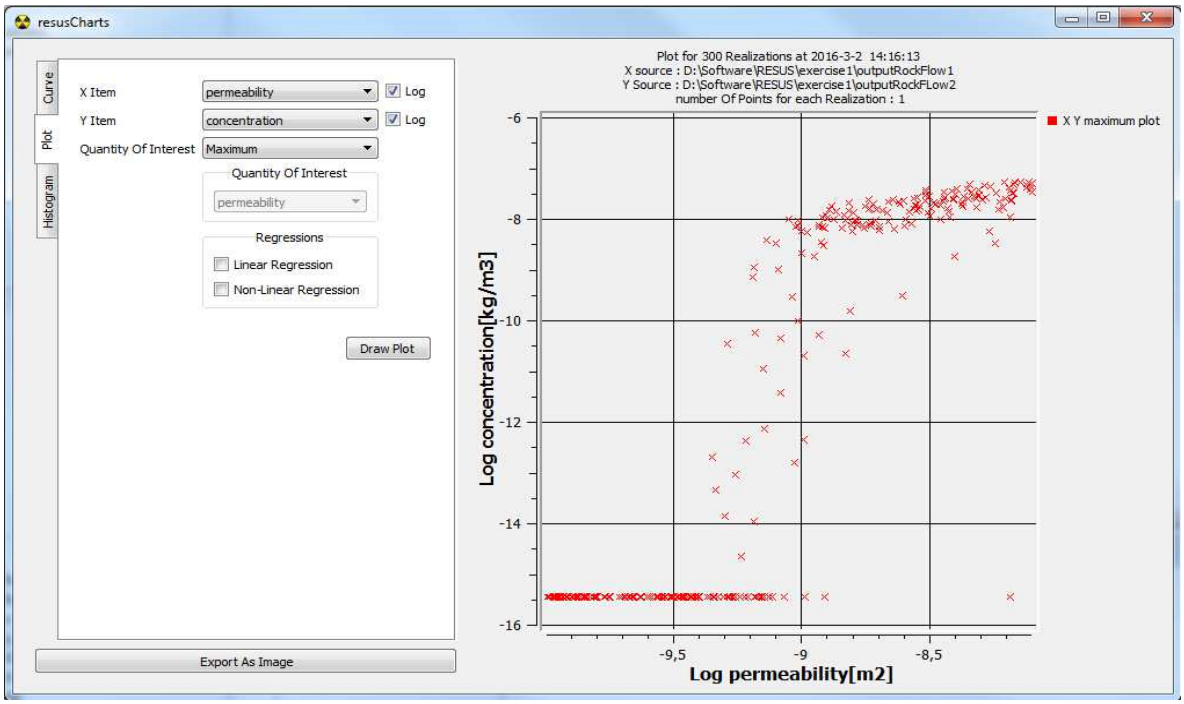


Figure 50. Scatter plot representing the corellation between input and output paramteres from two different layers of model

7.3.2 Parallelization

The ReSUS Platform can perform the simulations in a concurrent form by utilizing the concurrent running threads (for more information please refer to provided user handbook (Ghofrani & Li, 2016)). In parallelized mode, each assigned executable to the model will be executed with a thread which is assigned to it.

The RockFlow model presented in 7.3.1 consists of two components which can be executed parallel to each other. Therefore, the sum of two threads, one thread for each model component, is applied to perform the simulation in the parallelized mode.

At the end, the amount of time consumed for simulation in the single thread form is compared with the time which is needed to run the parallel simulation. The simulations are run five times to calculate an average simulation time, see Table 5.

Table 5. Parallelization of RockFlow Model by ReSUS Platform

Single mode (time in seconds)	Parallel mode (time in seconds)	Run number
3900	**2530**	**1**
3850	**2611**	**2**
4000	**2700**	**3**
3795	**2698**	**4**
3940	**2734**	**5**
Average:3933	**Average:2654**	

The calculated speedup shows a 48% improvement in simulation time.

$$Speedup = \frac{T_1}{T_p} = \frac{3933}{2654} = 1.48$$

These values are just a demonstration of the fact that parallelization by ReSUS Platform can improve the speedup factor. Hence, the presented speedup value is not a general statement about the parallelization factor of the ReSUS Platform.

7.3.3 Generating random values as input with MatLab (HDF5)

The purpose of this part is to show the potential of the ReSUS Platform in connecting to third-party random generator tools which allows using sampling methods other than the ones implemented in ReSUS. The MATLAB function (createParameterValuesInHDF5()) generates random sample values using the Mersenne Twister (Matsumoto & Nishimura, 1998) sampling method. This function then writes the sample values and their meta-data in HDF5 files. The structure of the created HDF5 files matches the accepted structure of the generated HDF5 by ReSUS.

```
function myoutput = createParameterValuesInHDF5(filname,length,
name,index,unit,distribution,logaritmic,min,max )

x = min + (max-min).*rand(length,1);

datasetname=strcat('/',name)

if strcmp(logaritmic,'True')==1

      LA = log(min)

    LB = log(max)

      x= exp(LA + (LB-LA) * rand(length,1))

end

h5create(filname,datasetname,[length 1])

h5write(filname,datasetname,x);

h5writeatt(filname,datasetname,'index',index);

h5writeatt(filname,datasetname,'name',name);

h5writeatt(filname,datasetname,'unit',unit);

h5writeatt(filname,datasetname,'distribution',distribution);

h5writeatt(filname,datasetname,'log',logaritmic);

h5writeatt(filname,datasetname,'min_value', num2str(min));

h5writeatt(filname,datasetname,'max_value', num2str(max));

h5writeatt(filname,datasetname,'show_in_output','True');

end
```

The commands in the section below call this function in order to generate random values for the four probabilistic input parameters of the Ishigami (Ishigami & Homma, 1990) model, which is introduced in 7.1.1 .

```
createParameterValuesInHDF5 ('hdf5generatedbymatlab.h5',3000,'param1','1', '-',
'Uniform', 'False',-3.1415,3.1415)

createParameterValuesInHDF5 ('hdf5generatedbymatlab.h5',3000,'param2','2''-',
'Uniform', 'False',-3.1415,3.1415)

createParameterValuesInHDF5 ('hdf5generatedbymatlab.h5',3000,'param3','3''-',
'Uniform', 'False',-3.1415,3.1415)

createParameterValuesInHDF5 ('hdf5generatedbymatlab.h5',3000,'param4','4','-',
'Uniform', 'False',-3.1415,3.1415)
```

In this step, the HDF5 file that is generated using the MatLab function above is replaced by the input files of designed simulation for Ishigami. Despite of different sources of the probabilistic input values, the estimated linear and non-linear regression by Chart Tool of ReSUS are obviously similar (Figure 51).

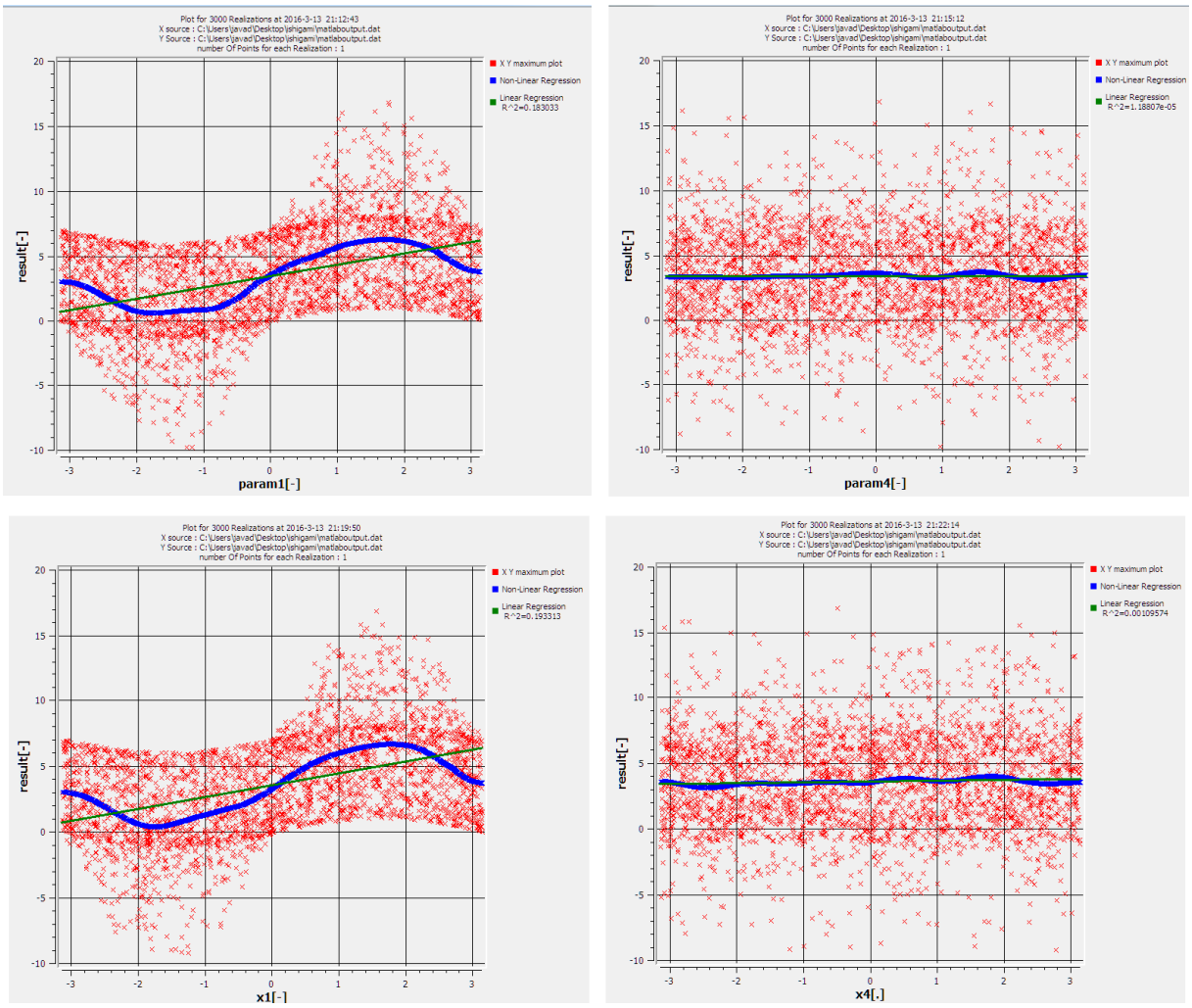


Figure 51. Charts Tool illustrates the results of Ishigami simulation with the generated inputs from MatLab (top) and Sampler Generator of ReSUS (down)

Moreover, the createParameterValuesInHDF5() function is used to generate the input samples for the Level-E simulation described in 7.1.2 by using the following set of commands in MatLab. It generates 300 random values for each probabilistic input parameter of the Level-E model which are mentioned in Table 2. The randomly generated values with this method are stored in a HDF5 formatted file named levelEInputMatlab.h5.

```
createParameterValuesInHDF5('leveEInputMatlab.h5',300,
'zeitdauer','1','unit','Uniform','False',100,1000)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'freisetzungrate von
Jod','2','unit','Uniform','True',0.001,0.01)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'freisetzungrate
Np','3','unit','Uniform','True',1.0E-6,1.0E-5)

createParameterValuesInHDF5('leveEInputMatlab.h5',300,
'abstandgeschwindikgeit1','4','unit','Uniform','True',0.001,0.1)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'length geschichte
1','5','unit','Uniform','False',100.0,500)

createParameterValuesInHDF5('leveEInputMatlab.h5',300,
'retardationsfaktor','6','','Uniform','False',1,5)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'retardationsfaktor
Np','7','unit','Uniform','False',3,30)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'abstandgeschwindigkeit
geoschicht2','8','unit','Uniform','True',0.01,0.1)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'length geoschichte
2','9','unit','Uniform','False',50.0,200)

createParameterValuesInHDF5('leveEInputMatlab.h5',300,
'retardationsfaktor2','10','unit','Uniform','False',1,5)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'retardationsfaktor
Geo.2','11','unit','Uniform','False',3.0,30.0)

createParameterValuesInHDF5('leveEInputMatlab.h5',300, 'betrachteten volumen von
wasser','12','unit','Uniform','True',100000.0,1.0E7)
```

Figure 52 demonstrates the generated plots with Chart Tool of ReSUS for simulations with the provided input values by Sample Generator of ReSUS (left) as well as the delivered random values by MatLab function (right). In both presented charts in Figure 52, the values of the input parameter with the index of 4 (water velocity L1/ Abstandgeschwindigkeit1) are plotted against the maximum of concentration (output parameter of Level-E simulation) values for each run of simulation.

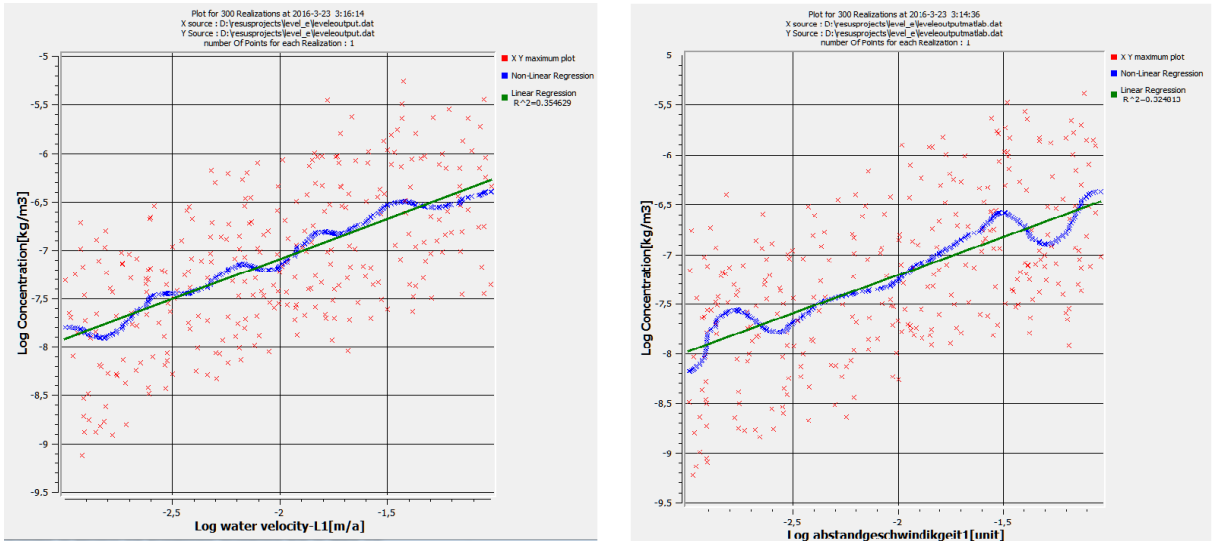


Figure 52. Comparison of the analyzing the input of the Level-E for index 4 against the calculated concentration values as output of the simulation model

The similarity between the visualized forms of the simulation parameters is not limited to the plot charts. Figure 53 illustrates the histograms generated from the randomly sampled values for fourth parameter by the Sample Generator of ReSUS in comparison with the values generated by the MatLab function.

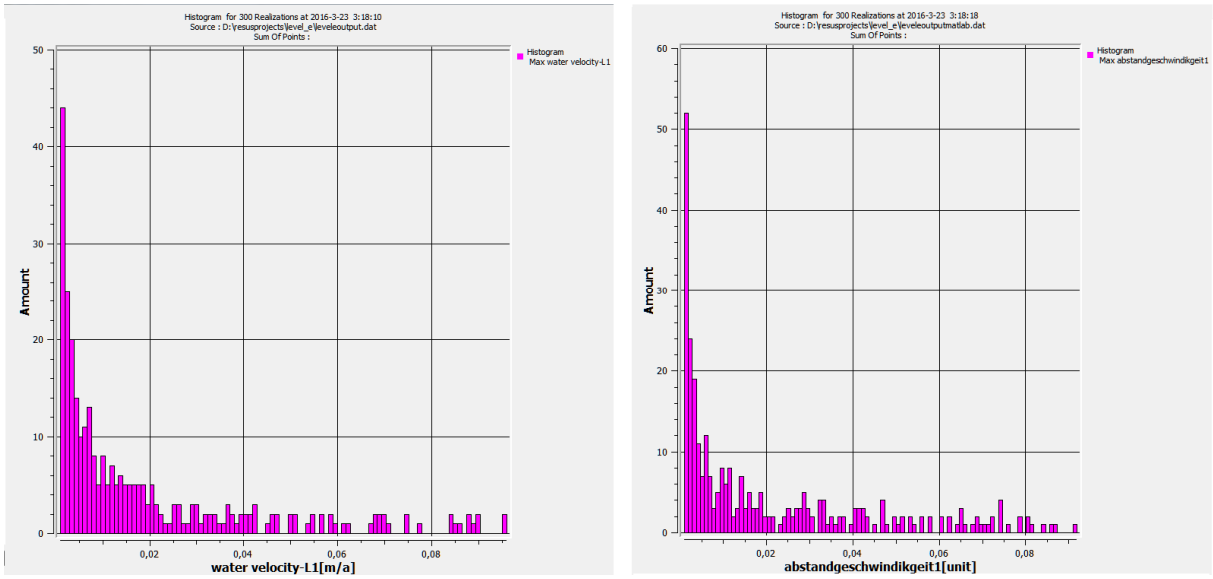


Figure 53. Histogram (drawn by Chart Tool) from the randomly generated values for same paramter with the Sample Generator of ReSUS (left) and a MatLab function (right)

Moreover, illustrating the output values of the simulations with the provided input values from different sampling methods shows the major similarities in the Histograms depicted in Figure 54.

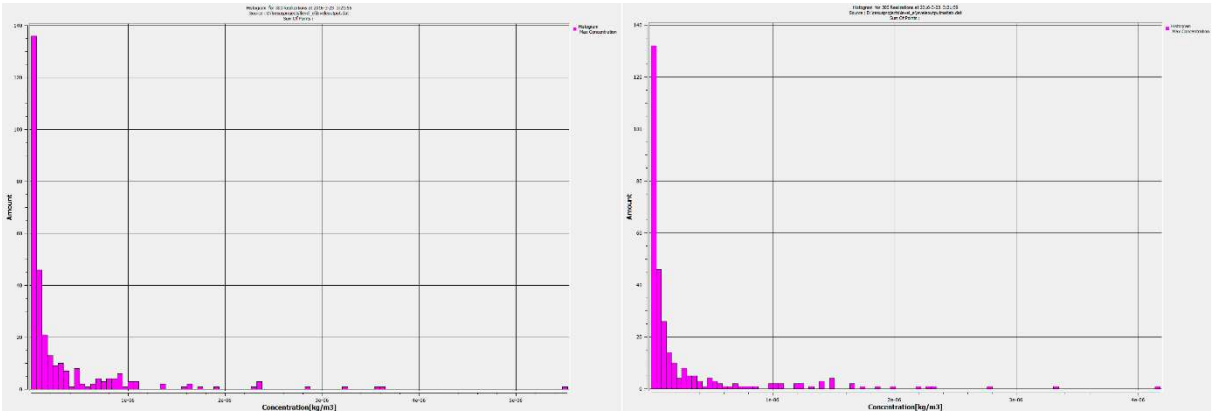


Figure 54. Histograms of the simulation outputs that are performed with the generated inputs with the ReSUS Sample Generator (left) and the MatLab function (right)

7.4 Summary

All the objectives of this chapter including the benchmarking, using the probabilistic simulation workflows and working with external probabilistic tools are fulfilled in the implemented example cases in a way that one can claim that the ReSUS platform is an eligible candidate to perform probabilistic simulations and to apply uncertainty and sensitivity analysis.

The examples presented in this chapter show that the ReSUS could utilize different types of simulation programs such as Level-E, TOUGH2, PHAST and simple executables that read a text file as input and generate some text files as output. The examples presented in the first part of this section are used for benchmarking purposes. Furthermore, the presented example in this section show the ability of ReSUS to use the input of simulations provided from third party input sampling tools using the HDF5 file format.

8 Conclusion and future work

Probabilistic safety assessment and sensitivity analysis involving numerical models for simulating thermal, hydraulic, mechanical chemical (THMC) and states and processes form an essential part of safety cases for deep (geologic) disposal. In this thesis, the conceptualization and development of a software platform for such analyses is described.

After having introduced some basic aspects of the analyses, a review of the state of the art is performed. The review, which addresses, inter alia, the predecessor of the platform presented here, studied to which extent features desirable for these analyses are implemented in existing software. It reveals that most of the codes considered offer the framework for probabilistic sensitivity analyses for single and simple simulation models using their embedded functions and libraries. Several of them also offer workflow management or a common data framework. Only a few, however, offer debugging features or library couplings. The abilities of running external THMC models and bringing them together into a simulation workflow are also seldom realized, which results in a lack of flexibility.

Therefore, the concept of the ReSUS platform (Repository Simulation, Uncertainty propagation and Sensitivity Analysis) was developed based on the ReSUS predecessor version. The concept includes an approach to use external deterministic simulators for studying the time-dependent systems in the safety assessment framework, which results in the ability to combine the power of standalone deterministic simulation programs with the ability to perform probabilistic simulations and to study uncertainty and sensitivity. ReSUS is able to work with parallelized simulations supported by multi-threading, which leads to a shorter execution time. ReSUS is meant to be used both in research and education contexts. Thus, ReSUS covers most of the features required for probabilistic safety assessment and sensitivity analysis.

Additional embedded features include result management, model consistency check and execution management. However, ReSUS is not able to couple the DLLs directly. In order to use the provided functions in a DLL by the ReSUS Platform, the DLL should be assigned to a code and its compiled executable can be used in ReSUS. This point requires programming knowledge in other software tools, too.

The thesis describes the software development process, starting by analyzing the requirements and continuing by describing the product perspective and the logical design of the components of the ReSUS platform. This is followed by a description of the implementation of the different components of the platform which can be used as a guide for further development of ReSUS, as well as tests and the deployment details. Furthermore, a user guide is prepared to help the user to make an easy begin with the ReSUS platform. It contains some typical examples and case studies. The reliability and flexibility of the ReSUS Platform is shown by implementing and benchmarking several simulation examples, in particular probabilistic simulations including uncertainty and sensitivity analysis.

Although the ReSUS Platform enables deterministic simulation tools to be applied to probabilistic approaches, several problems are still open in particular when considering assigning new simulator programs to ReSUS. Currently only programs using ASCII input files can be assigned to the ReSUS Platform. Deterministic simulation programs which use binary or GUI-based formats are not yet supported by the ReSUS Platform. Furthermore, ReSUS is limited to work with executables under the Windows operation system.

Based on the ReSUS version presented in this thesis, further development of the ReSUS Platform in several areas is conceivable:

1- Assigning more deterministic simulation programs including programs from fields other than radioactive waste disposal

2- Creating the possibility of assigning external programs with types of inputs other than ASCII input format

3- Enabling the ReSUS Platform to work with programs that are compatible with other operation system such as Linux

4- Enabling the ReSUS Platform to work with distributed environments (web services or distributed operation systems)

5- Extending the ReSUS platform with a library of templates for typical assessment situations

6- Developing a cloud based version of ReSUS

7- Implementing sampling methods (e. g. Latin Hypercube Sampling) as well as additional sensitivity analysis methods such as first order effect, moment independent, imperative measures

8- Accounting for research results concerning the handling of time series

9- Handling of input uncertainties with correlation or dependency structures

9 Appendix

This Appendix includes the XML files which describe the simulation workflows aforementioned in section 7. Furthermore, the Template files used to generate the input of the simulations is attached.

9.1 RockFlow

9.1.1 Model

```
<?xml version="1.0" encoding="UTF-8"?><simulation>
  <threads>1</threads>
  <model>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>241</x>
      <y>121</y>
    </location>
    <id>19</id>
    <name>Rockflow1 model</name>
    <modelType>first rockflow model</modelType>
    <description/>
    <executor>rf5110_win32.exe</executor>
    <timeout>0</timeout>
    <breakIfNotZero>false</breakIfNotZero>
    <workingDirectory>D:\RockFlowModel</workingDirectory>
    <executionParameters>tangv1</executionParameters>
    <numberOfInputPins>1</numberOfInputPins>
    <numberOfOutputPins>1</numberOfOutputPins>
    <inputFileName>tangV1.rfd</inputFileName>
    <outputFile>
      <fileName>tangv1.plt</fileName>
      <numberOfLines/>
      <minFileSize/>
      <breakIfHappend>false</breakIfHappend>
    </outputFile>
    <logFile>
      <fileName/>
      <numberOfLines/>
      <breakIfHappend>false</breakIfHappend>
    </logFile>
  </model>
  <connection>
    <source>
      <id>18</id>
      <pin>OUT</pin>
    </source>
    <target>
      <id>19</id>
      <pin>1</pin>
    </target>
  </connection>
  <model>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>727</x>
      <y>161</y>
```

```
    </location>
    <id>23</id>
    <name>Rockflow2 model</name>
    <modelType>rockflow model</modelType>
    <description/>
    <executor>rf5110_win32.exe</executor>
    <timeout>0</timeout>
    <breakIfNotZero>false</breakIfNotZero>
    <workingDirectory>D:\RockFlowModel</workingDirectory>
    <executionParameters>asm2d</executionParameters>
    <numberOfInputPins>2</numberOfInputPins>
    <numberOfOutputPins>1</numberOfOutputPins>
    <inputFileName>asm2d.rfd</inputFileName>
    <outputFile>
      <fileName>asm2d.plt</fileName>
      <numberOfLines/>
      <minFileSize/>
      <breakIfHappend>false</breakIfHappend>
    </outputFile>
    <logFile>
      <fileName/>
      <numberOfLines/>
      <breakIfHappend>false</breakIfHappend>
    </logFile>
  </model>
  <connection>
    <source>
      <id>22</id>
      <pin>OUT</pin>
    </source>
    <target>
      <id>23</id>
      <pin>2</pin>
    </target>
  </connection>
  <connection>
    <source>
      <id>21</id>
      <pin>OUT</pin>
    </source>
    <target>
      <id>23</id>
      <pin>1</pin>
    </target>
  </connection>
  <resultConverter>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>411</x>
      <y>139</y>
    </location>
    <id>20</id>
    <name>RC20</name>
    <numberOfInputPins>1</numberOfInputPins>
    <regex>[[:space:]]*[-+]?[0-9]+ [[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-
+]?[0-9]+)? [[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?
[[:space:]]*</regex>
    <columnDelimiter>[[:space:]]</columnDelimiter>
    <executor>
      <fileName/>
```

```
      <workingDirecotry/>
      <executorParameter/>
      <executorOutputFileName/>
    </executor>
    <inputFileName>D:\RockFlowModel\tangv1.plt</inputFileName>
    <outputFileName>D:\RockFlowModel\rockflow1output.dat</outputFileName>
    <numberOfOutputRows>0</numberOfOutputRows>
    <outputAffectedIndex>-1</outputAffectedIndex>
    <index>
      <id>1</id>
      <tag>time</tag>
      <unit>year</unit>
      <forward>true</forward>
      <coefficient>1.0</coefficient>
    </index>
    <index>
      <id>2</id>
      <tag>conc1</tag>
      <unit>kg/m3</unit>
      <forward>true</forward>
      <coefficient>1.0</coefficient>
    </index>
  </resultConverter>
  <connection>
    <source>
      <id>19</id>
      <pin>O1</pin>
    </source>
    <target>
      <id>20</id>
      <pin>1</pin>
    </target>
  </connection>
  <resultConverter>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>902</x>
      <y>211</y>
    </location>
    <id>24</id>
    <name>RC24</name>
    <numberOfInputPins>1</numberOfInputPins>
    <regex>[[:space:]]*[-+]?[0-9]+ [[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-
+]?[0-9]+)? [[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?
[[:space:]]*</regex>
    <columnDelimiter>[[:space:]]</columnDelimiter>
    <executor>
      <fileName/>
      <workingDirecotry/>
      <executorParameter/>
      <executorOutputFileName/>
    </executor>
    <inputFileName>D:\RockFlowModel\asm2d.plt</inputFileName>
    <outputFileName>D:\RockFlowModel\rockflow2output.dat</outputFileName>
    <numberOfOutputRows>0</numberOfOutputRows>
    <outputAffectedIndex>-1</outputAffectedIndex>
    <index>
      <id>1</id>
      <tag>time2</tag>
      <unit>s</unit>
```

```
      <forward>true</forward>
      <coefficient>1.0</coefficient>
    </index>
    <index>
      <id>2</id>
      <tag>con2</tag>
      <unit>kg/m3</unit>
      <forward>true</forward>
      <coefficient>1.0</coefficient>
    </index>
  </resultConverter>
  <connection>
    <source>
      <id>23</id>
      <pin>O1</pin>
    </source>
    <target>
      <id>24</id>
      <pin>1</pin>
    </target>
  </connection>
  <IOProvider>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>44</x>
      <y>71</y>
    </location>
    <id>18</id>
    <numberOfInputPins>1</numberOfInputPins>
    <logFile/>
    <index/>

<parametersFileName>D:\RockFlowModel\rockflow1inputs.h5</parametersFileName
>
  </IOProvider>
  <IOProvider>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>567</x>
      <y>143</y>
    </location>
    <id>21</id>
    <numberOfInputPins>1</numberOfInputPins>
    <logFile/>
    <index>1</index>
  </IOProvider>
  <connection>
    <source>
      <id>20</id>
      <pin>OUT</pin>
    </source>
    <target>
      <id>21</id>
      <pin>1</pin>
    </target>
  </connection>
  <IOProvider>
```

```
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>572</x>
      <y>309</y>
    </location>
    <id>22</id>
    <numberOfInputPins>1</numberOfInputPins>
    <logFile/>
    <index/>

<parametersFileName>D:\RockFlowModel\rockflow2inputs.h5</parametersFileName
>
  </IOProvider>
</simulation>
```

9.1.2 Template for input files of the first RockFlow model (tangV1.rfd.tmp)

```
; RockFlow 3.8.35
; Tutorial B2 Version 0.1.0

#PROJECT
  TANG Variante 1 (Matrixdiffusion, Einleitung und Simulation über 6 Tage)



#MODEL
 1      ; simulation flag
 10097  ; model identifier
 0      ; flow model flag
 0      ; convection model flag
 1      ; chemical model flag
 0      ; transport phase of multiphase model
 1      ; simulation optimizer flag
 2      ; material groups
 1      ; phases
 1      ; components
 0      ; adaptive mesh refinement flag
 0      ; chain_reaction_model
 0      ; heat_reaction_model
 0      ; saturation_calculation_method
 0      ; mobile immobile model flag

#TIME
 0.0    ; final simulation time
 0      ; maximum time step number
 0      ; time step control
 1000   ; time step number
 9460800000  ; time step length

#OUTPUT
 0       ; files
 0       ; geometry
 0       ; initial condition
 0       ; format
 1       ; numbering
 3       ; type
```

```
 9460800000 ; parameters

#OUTPUT_EX
 11                  ; type
 tangv1.plt          ; name
 1000 1     0        ;  x, y, z
 1                   ; mode
 1                   ; method
 0                   ; data output_method
 9460800000             ; time
 1.0               ; time radius
 2                   ; number of variables
 X      CONC         ;    kg/m3   Concentration_of_nuclide_1_at_model_1
1



#NUMERICS
 0         ; numerical method
 PRESSURE  ; name of unknown variable
 2         ; gauss points
 1.0       ; time collocation
 0.0       ; upwind parameter
 0  0.0    ; lagrange methode, quality parameter

#NUMERICS
 1         ; numerical method
 TRANSPORT ; name of unknown variable
 2         ; gauss points
 0.5       ; time collocation
 0.0       ; upwind parameter
 0  0.0    ; lagrange methode, quality parameter


#NUMERICS
 1               ; numerical method
 TRANSPORT_SORP ; name of unknown variable
 2               ; gaussian points
 0.5             ; time collocation
 0.0             ; upwind parameter
 0               ; lagrange methode
 0.0             ; quality parameter


#LINEAR_SOLVER_PROPERTIES_PRESSURE
2        ; method
0        ; norm
1        ; preconditioning
10000    ; maximum iterations
0        ; repeating
1        ; criterium
1.0e-09 ; absolute error
0        ; kind
2        ; matrix storage technique

#LINEAR_SOLVER_PROPERTIES_CONCENTRATION
2        ; method
0        ; norm
1        ; preconditioning
10000      ; maximum iterations
0        ; repeating
1        ; criterium
```

```
1.0e-010 ; absolute error
0        ; kind
2        ; matrix storage technique

#LINEAR_SOLVER_PROPERTIES_SORBED_CONCENTRATION
2        ; method
0        ; norm
1        ; preconditioning
10000      ; maximum iterations
0        ; repeating
1        ; criterium
1.0e-010 ; absolute error
0        ; kind
2        ; matrix storage technique

#BOUNDARY_CONDITIONS_PRESSURE
1               ; type
0               ; mode
0               ; curve
0.0  1.0  0.0   ; x,y,z
0.001           ; radius
200000.0        ; value

1               ; type
0               ; mode
0               ; curve
1000.0  1.0  0.0  ; x,y,z
0.001           ; radius
199000.0        ; value

#BOUNDARY_CONDITIONS_CONCENTRATION
1               ; type
0               ; mode
1               ; curve
0.0  1.0  0.0   ; x,y,z
0.001           ;
5.66e-08             ; value


#CURVES
0     1.0
3.1536e11   1.0
3.1537e11   0.0
9.4608e12   0.0


#FLUID_PROPERTIES
0  1000.0   ; density function, parameter
0  0.001    ; viscosity function, parameter
0.0         ; real gas factor
0.0  0.0    ; heat capacity, heat conductivity


#SOIL_PROPERTIES
 1                                ; dimension
 0.6e-4                           ; area
 0                                ; porosity model

 1.0                             ;   porosity_fracture    -
porosity_of_fracture_in_granite    1   uniform    0.8     1.0
 ;0.0                             ; tortuosity
 1.0                              ; tortuosity
 0                                ; mobile immobile model
```

```
 0                                  ; lithological component
 0                                  ; maximum sorption model
 0                                  ; nonlinear flow parameter
 0.0                                ; storativity
 0  0                               ; permeability model, permeability
tensor
;SIMULATE
 <$1>                           ; permeability    m2
permeability of fracture   1   log uniform    1.e-10     1.e-8
 0.0  0.0  0.0  0.0  0.0  0.0  0.0 ; k-S function
 0.0  0.0  0.0  0.0  0.0  0.0  0.0 ; p-S function
 0.76  0.0                          ; mass dispersion parameters
 0.0   0.0                          ; heat dispersion parameters
 0.0  0.0                           ; rock density, heat capacity
 0  0.0                            ; heat conductivity parameters

#SOIL_PROPERTIES
; matrix
 2                                  ; dimension
 1.0                                ; area
 0                                  ; porosity model
;SIMULATE
 0.2                               ; porosity_matrix   -
porosity_of_matrix    1   uniform       0.2     0.5
 1.0                                ; tortuosity
 ;0.0                               ; tortuosity
 0                                  ; mobile immobile model
 0                                  ; lithological component
 0                                  ; maximum sorption model
 0                                  ; nonlinear flow parameter
 0.0                                ; storativity
 0  0                               ; permeability model, permeability
tensor
 1.0e-099                           ; permeability
 0.0  0.0  0.0  0.0  0.0  0.0  0.0 ; k-S function
 0.0  0.0  0.0  0.0  0.0  0.0  0.0 ; p-S function
 0.0   0.0                          ; mass dispersion parameters
 0.0   0.0                          ; heat dispersion parameters
 0.0  0.0                           ; rock density, heat capacity
 0  0.0                            ; heat conductivity parameters

#COMPONENT_PROPERTIES
1

<$2>      ;     diffusion      m2/s
MatrixDiffusion_coefficient_of_solution     1     uniform      1.e-13
1.e-10
1    1.05e-14          ; decay_sol model type, lambda_sol [1/s]
1    1.05e-14          ; decay_sorp model type, lamda_sorp [1/s]
1
;SIMULATE
<$3>          ;   kd_fracture    kg/kg     kd_of_layer1   1   uniform
0.00005    0.0005
0          ; chemical nonequilibrium model type
0          ; physical nonequilibrium model type
0          ; model type, solution dependence type

#STOP
```

9.1.3 Template for input files of second RockFlow model (asm2d.rfd.tmp)

```
; RockFlow 3.8.35
; Tutorial A1 Version 0.1.0

#PROJECT
  ASM 2D (2D STROEMUNGSMODELL)

#MODEL
 1      ; simulation flag
 10097  ; model identifier
 0      ; flow model flag
 0      ; convection model flag
 1      ; chemical model flag
 0      ; transport phase of multiphase model
 2      ; simulation optimizer flag
 1      ; material groups
 1      ; phases
 1      ; components
 0      ; adaptive mesh refinement flag
 0      ; chain_reaction_model
 0      ; heat_reaction_model
 0      ; saturation_calculation_method
 0      ; mobile immobile model flag

#TIME
 0.0    ; final simulation time
 0      ; maximum time step number
 0      ; time step control
 1000   ; time step number
 9460800000  ; time step length

#OUTPUT_EX
 11                   ; type
 asm2d.plt           ; name
 1000 3     0         ;  x, y, z
 1                    ; mode
 1                    ; method
 0                    ; data_output_method
 9460800000             ; time
 1.0                ; time radius
 2                    ; number of variables
 X      CONC          ;    kg/m3   Concentration_of_nuclide_1_at_model_2
1




#OUTPUT
 0    ; files
 0    ; geometry
 0    ; initial condition
 0    ; format
 1    ; numbering
 3    ; type
 1.0  ; parameters

#NUMERICS
 0        ; numerical method
 PRESSURE ; name of unknown variable
 2        ; gauss points
```

```
 1.0      ; time collocation
 0.0      ; upwind parameter
 0        ; lagrange methode
 0.0      ; quality parameter

#NUMERICS
 1         ; numerical method
 TRANSPORT ; name of unknown variable
 2         ; gauss points
 0.5       ; time collocation
 0.0       ; upwind parameter
 0  0.0    ; lagrange methode, quality parameter

#NUMERICS
 1              ; numerical method
 TRANSPORT_SORP ; name of unknown variable
 2              ; gaussian points
 0.5            ; time collocation
 0.0            ; upwind parameter
 0              ; lagrange methode
 0.0            ; quality parameter



#LINEAR_SOLVER_PROPERTIES_CONCENTRATION
2        ; method
0        ; norm
1        ; preconditioning
100000      ; maximum iterations
0        ; repeating
1        ; criterium
1.0e-014 ; absolute error
0        ; kind
2        ; matrix storage technique

#LINEAR_SOLVER_PROPERTIES_PRESSURE
 2     ; method
 0     ; norm
 1     ; preconditoning
 100000     ;maximum interations
 1     ; repeating
 1     ; criterium
 1e-012     ;absolute error
 0     ; kind
 2     ; matrix storage technique

#LINEAR_SOLVER_PROPERTIES_PRESSURE
 2     ; method
 0     ; norm
 1     ; preconditoning
 100000     ;maximum interations
 1     ; repeating
 1     ; criterium
 1e-012     ;absolute error
 0     ; kind
 2     ; matrix storage technique




#BOUNDARY_CONDITIONS_CONCENTRATION
 1              ; type
 0              ; mode
```

```
 1               ; curve
 0.0  0.0  0.0 ; x1,y1,z1
 0.001           ; radius
 1.0             ; value1
#CURVES
<$1>



#BOUNDARY_CONDITIONS_PRESSURE
 2               ; type
 0               ; mode
 0               ; curve
 1000.0  0.0  0.0 ; x0,y0,z0
 1000.0  3.0  0.0 ; x1,y1,z1
 0.001           ; radius
 4890000.0         ;   Pressure     Pa     hydraulic_pressure_of_right_side
1     uniform      100     1000
 4890000.0         ;   Pressure     Pa     hydraulic_pressure_of_right_side
1     uniform      100     1000


 2               ; type
 0               ; mode
 0               ; curve
 0.0  0.0  0.0 ; x0,y0,z0
 0.0  3.0  0.0 ; x1,y1,z1
 0.001           ; radius
 4900000.0         ; value0
 4900000.0         ; value1



#FLUID_PROPERTIES
 0  1000.0 ; density function, parameter
 0  0.001  ; viscosity function, parameter
 0.0       ; real gas factor
 0.0  0.0  ; heat capacity, heat conductivity

#COMPONENT_PROPERTIES
 1      1.0e-8                  ; diffusion    m2/s
MatrixDiffusion_coefficient_of_solution
 1    1.05e-14                ; decay_sol model type, lambda_sol [1/s],
ratio
 1    1.05e-14             ; decay_sorp model type, lambda_sop [1/s], ratio
 1
;SIMULATE
 <$2>            ; kd_layer    kg/kg     kd_of_layer1   1   uniform
0.00005    0.0005
 0               ; chemical nonequilibrium model type
 0               ; physical nonequilibrium model type
 0               ; model type, solution dependence type



#SOIL_PROPERTIES
 2                               ; dimension
 1.0                             ; area
 0                               ; porosity model
;SIMULATE
 <$3>                             ;   porosity    -
porosity_of_layer1    1   uniform       0.05     0.4
```

```
 1.0                                ; tortuosity
 0                                  ; mobile immobile model
 0                                  ; lithological component
 0                                  ; maximum sorption model
 0                                  ; nonlinear flow parameter
 0.0                                ; storativity
 0  0                               ; permeability model, permeability
tensor
;SIMULATE
<$4>                            ; permeability  -   permeability_of_layer1  1
log_uniform    1.e-13   1.e-11
 0.0  0.0  0.0  0.0  0.0  0.0  0.0 ; k-S function
 0.0  0.0  0.0  0.0  0.0  0.0  0.0 ; p-S function
;SIMULATE
<$5>                            ;  Dispersionlänge     m
Dispersion_of_layer2   1    uniform     5     10
 0.0                              ; mass dispersion parameters
 0.0  0.0                             ; heat dispersion parameters
 2000.0  0.0                          ; rock density, heat capacity
 0  0.0                              ; heat conductivity parameters



#STOP
```

9.2 PHAST model

9.2.1 Model

```
<?xml version="1.0" encoding="UTF-8"?><simulation>
  <threads>1</threads>
  <IOProvider>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>195</x>
      <y>280</y>
    </location>
    <id>35</id>
    <numberOfInputPins>1</numberOfInputPins>
    <logFile/>
    <index/>
    <parametersFileName>D:\phastResus\Szenario_1\phastsampleinput.h5</parametersFileName>
  </IOProvider>
  <model>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>411</x>
      <y>331</y>
    </location>
    <id>36</id>
    <name>phast model</name>
    <modelType/>
    <description/>
    <executor>phast.bat</executor>
    <timeout>0</timeout>
    <breakIfNotZero>false</breakIfNotZero>
    <workingDirectory>D:\phastResus\Szenario_1</workingDirectory>
    <executionParameters>100_10 wateq4f.dat</executionParameters>
    <numberOfInputPins>1</numberOfInputPins>
    <numberOfOutputPins>1</numberOfOutputPins>
    <inputFileName>100_10.trans.dat</inputFileName>
    <outputFile>
      <fileName>100_10.chem.xyz.tsv</fileName>
      <numberOfLines/>
```

```
      <minFileSize/>
      <breakIfHappend>false</breakIfHappend>
    </outputFile>
    <logFile>
      <fileName/>
      <numberOfLines/>
      <breakIfHappend>false</breakIfHappend>
    </logFile>
  </model>
  <connection>
    <source>
      <id>35</id>
      <pin>OUT</pin>
    </source>
    <target>
      <id>36</id>
      <pin>1</pin>
    </target>
  </connection>
  <resultConverter>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>701</x>
      <y>362</y>
    </location>
    <id>37</id>
    <name>RC37</name>
    <numberOfInputPins>1</numberOfInputPins>
    <regex>[[:space:]]*[-+]?[0-9]+[[:space:]]*[-+]?[0-9]+[[:space:]]*[-+]?[0-9]+[[:space:]]*[-
+]?[0-9]+[[:space:]]*[-+]?[0-9]+[[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-
9]+)?[[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?[[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-
+]?[0-9]+)?[[:space:]]*[-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?[[:space:]]*[-+]?[0-9]*\.?[0-
9]+([eE][-+]?[0-9]+)?[[:space:]]</regex>
    <columnDelimiter>[[:space:]]</columnDelimiter>
    <executor>
      <fileName/>
      <workingDirecotry/>
      <executorParameter/>
      <executorOutputFileName/>
    </executor>
    <inputFileName>D:\phastResus\Szenario_1\100_10.chem.xyz.tsv</inputFileName>
    <outputFileName>D:\phastResus\output.dat</outputFileName>
    <numberOfOutputRows>0</numberOfOutputRows>
    <outputAffectedIndex>-1</outputAffectedIndex>
    <index>
      <id>9</id>
      <tag>U</tag>
      <unit>mol/liter</unit>
      <forward>true</forward>
      <coefficient>1.0</coefficient>
    </index>
    <index>
      <id>3</id>
      <tag>time</tag>
      <unit>year</unit>
      <forward>true</forward>
      <coefficient>1.0</coefficient>
    </index>
  </resultConverter>
  <connection>
     <source>
      <id>36</id>
      <pin>O1</pin>
    </source>
    <target>
      <id>37</id>
      <pin>1</pin>
    </target>
  </connection>
</simulation>
```

9.2.2 Phast.bat

```
@echo off
setlocal
```

```
set TD=%~dp0

:INPUT
"%TD%\phastinput.exe" "%~1" "%~2"
IF NOT ERRORLEVEL 1 GOTO RUN
GOTO END

:RUN
"%TD%\phast-ser.exe"

:END

Endlocal
```

9.2.3 Template (100_10.trans.dat)

```
SOLUTE TRANSPORT True
        -diffusivity 1e-009
STEADY FLOW true
        -head_tolerance 1e-005
        -flow_balance_tolerance 0.001
        -minimum time step  0.001
        -maximum time step 0.1
        -iterations 1000
        -growth factor 2
FREE_SURFACE_BC True
SOLUTION_METHOD
        -iterative solver   true
        -tolerance          1e-08
        -save directions    20
        -maximum_iterations 2000
        -space_differencing 0.0
        -time differencing  1.0
        -cross dispersion   false
        -rebalance fraction 0.5
        -rebalance_by_cell  false
UNITS
        -time                               years
        -horizontal grid                    m
        -vertical grid                      m
        -map horizontal                     m
        -map vertical                       m
        -head                               m
        -hydraulic_conductivity             m/s
        -specific storage                   1/m
        -dispersivity                       m
        -flux                               meters/years
        -leaky hydraulic conductivity       m/s
        -leaky_thickness                    m
        -well_diameter                      m
        -well flow rate                     m^3/years
        -well depth                         m
        -river bed hydraulic conductivity m/s
        -river_bed_thickness                m
        -river_width                        m
        -river depth                        m
        -drain hydraulic conductivity       m/s
        -drain thickness                    m
        -drain width                        m
        -equilibrium_phases                 WATER
        -exchange                           WATER
        -surface                            WATER
        -solid solutions                    WATER
        -kinetics                           WATER
        -gas phase                          WATER
GRID
        -uniform X 0 10000 101
        -uniform Y 0  100  11
        -uniform Z 0  400  21
        -chemistry dimensions XYZ
        -print orientation XY
        -grid_origin   0  0  0
        -grid_angle    0
MEDIA
        -box    0 0 0 10000 100 400
                -active                  1
```

```
              -Kx                       0.0001 # Spannweite: 0.01 - 1*10-xy;
Gesteinsabhängig. Kies, Sand, Schluff, Ton? Was genau eingrenzen?
              -Ky                       0.0001     #
              -Kz                       0.0001     #
              -porosity                 0.2 # Spannweite: 0.05 - 0.4
              -specific_storage         0
              -long_dispersivity        50  # 10 % der Diskretisierung
              -horizontal_dispersivity  5   # 10 % der Diskretisierung
              -vertical dispersivity    19  # 10 % der Diskretisierung
              -tortuosity               0   #hängt vom Gestein ab

FLUX_BC

-box 0 0 400 10000 100 400 #Niederschlag
              -face Z
              -flux
                     0      years -0.125
                     1      years -0.125   #GWN von -0.0 bei Permafrost bis -0.5? /zeitlich
konstant oder variabel?
              -associated_solution
                     0      years  10
                     1      years  10     #GW - roh - Vorgaben: Landwirtschaft, Weide, Wald,
Siedlung oder nur Niederschlag?


              #Quelle1
              -zone 1000 40 0 2000 60 0     #Fläche der Quelle 1000 m * 20 m
              -face Z
              -flux
                     0      years <$1>
                     #1     years 0.005           #Quellstärke 365 m³ /a = 365 m/(m² a) =
0.0365 m / (10000 m² a) - Spannweite benötigt
              -associated solution
                     0      years  100
                     #1     years  100    #Tracer - Wie zusammensetzung? Konstant Variabel?

HEAD_IC
              -domain
              -head                     y 400.0 0 350.0 100


CHEMISTRY_IC   # homogener chemischer aufbau oder herterogen? hier homogen

       -box 0 0 200 10000 100 400

              -solution             11
              -equilibrium_phases   10
              #-surface             10
              #-exchange            10
              #-kinetics            10


       -box 0 0 0 10000 100 200
              -solution             11
              -equilibrium phases   20
              #-surface             10
              #-exchange            10
              #-kinetics            10


RIVER 1
       -xy coordinate system GRID
       -z_coordinate_system  GRID
       -point 10000 0               # Flussverlauf kären! Ist die begrenzung des
einzugsgebiet.
              -head
                     0 years     380
                     1 years     380
              -solution
                     0 years     60
                     1 years     60
              -width                        100
              -bed hydraulic conductivity   0.001
              -bed thickness                1
              -depth                        40
       -point 10000 100
              -head
                     0 years     380
```

```
                1 years     380
            -solution
                0 years     60
                1 years     60
            -width                          100
            -bed_hydraulic_conductivity  0.001
            -bed_thickness               1
            -depth                          40

PRINT INITIAL
        -boundary_conditions        false
        -components                 false
        -conductances               false
        -echo_input                 true
        -fluid properties           true
        -force chemistry print      false
        -HDF_chemistry              true
        -HDF_heads                  true
        -HDF_media                  true
        -HDF_steady_flow_velocities true
        -heads                      true
        -media properties           false
        -solution_method            true
        -steady_flow_velocities     false
        -wells                      true
        -xyz_chemistry              true
        -xyz components             false
        -xyz_heads                  false
        -xyz_steady_flow_velocities false
        -xyz_wells                  false
PRINT_FREQUENCY
        -save_final_heads false
        0
            -bc_flow_rates          0
            -boundary_conditions    0
            -components             0
            -conductances           0
            -end of period default  true
            -flow balance           end
            #-force_chemistry_print  1    years
            -HDF_chemistry          100  years
            -HDF_heads              end
            -HDF_velocities         end
            -heads                  end
            -progress statistics    end
            -restart_file           0
            -velocities             end
            -wells                  end
            -xyz_chemistry          100  years
            -xyz components         0
            -xyz_heads              end
            -xyz_velocities         end
            -xyz_wells              end
            -zone_flow              end
            #-zone flow xyzt         10   years
            -zone flow tsv          end
            #-hdf_intermediate       10    years
PRINT_LOCATIONS
        -xyz_chemistry
            -box    0 0 0 10000 100 400
        -print 0
            -box 1000 80 0 1100 100 0
        -print 0
            -box 9900 40 380 9900 40 380
        -print 1

TIME CONTROL
        -time step    0        100 years
        -time_change  1000 years
        -start_time 0 years
END
```

9.3 TOUGH2

9.3.1 Model

```
<?xml version="1.0" encoding="UTF-8"?><simulation>
  <threads>1</threads>
  <model>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>311</x>
      <y>224</y>
    </location>
    <id>2</id>
    <name>Tough2-example</name>
    <modelType>tough2</modelType>
    <description/>
    <executor>execTough.bat</executor>
    <timeout>0</timeout>
    <breakIfNotZero>false</breakIfNotZero>
    <workingDirectory>D:\Software\TOUGH2\RESUS-Example - Kopie</workingDirectory>
    <executionParameters/>
    <numberOfInputPins>1</numberOfInputPins>
    <numberOfOutputPins>1</numberOfOutputPins>
    <inputFileName>EINGABE</inputFileName>
    <outputFile>
      <fileName>Plot_Ur235</fileName>
      <numberOfLines/>
      <minFileSize/>
      <breakIfHappend>false</breakIfHappend>
    </outputFile>
    <logFile>
      <fileName/>
      <numberOfLines/>
      <breakIfHappend>false</breakIfHappend>
    </logFile>
  </model>
  <connection>
    <source>
      <id>1</id>
      <pin>OUT</pin>
    </source>
    <target>
      <id>2</id>
      <pin>1</pin>
    </target>
  </connection>
  <resultConverter>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>517</x>
      <y>226</y>
    </location>
    <id>3</id>
    <name>RC3</name>
    <numberOfInputPins>1</numberOfInputPins>
    <regex>([-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-9]+)?)(\s+)([-+]?[0-9]*\.?[0-9]+([eE][-+]?[0-
9]+)?)</regex>
    <columnDelimiter>[[:space:]]</columnDelimiter>
    <executor>
      <fileName>t2t.bat</fileName>
      <workingDirecotry>D:\Software\TOUGH2\RESUS-Example - Kopie</workingDirecotry>
      <executorParameter/>
      <executorOutputFileName>t2tout.txt</executorOutputFileName>
    </executor>
    <inputFileName>D:\Software\TOUGH2\RESUS-Example - Kopie\Plot_Ur235</inputFileName>
    <outputFileName>D:\Software\TOUGH2\RESUS-Example - Kopie\outputtest.txt</outputFileName>
    <numberOfOutputRows>0</numberOfOutputRows>
    <outputAffectedIndex>-1</outputAffectedIndex>
    <index>
      <id>0</id>
```

```
      <tag>time</tag>
      <unit>year</unit>
      <forward>false</forward>
      <coefficient>3.2E-8</coefficient>
    </index>
    <index>
      <id>1</id>
      <tag>concentration</tag>
      <unit>kg/m3</unit>
      <forward>false</forward>
      <coefficient>1.0</coefficient>
    </index>
  </resultConverter>
  <connection>
    <source>
      <id>2</id>
      <pin>O1</pin>
    </source>
    <target>
      <id>3</id>
      <pin>1</pin>
    </target>
  </connection>
  <IOProvider>
    <size>
      <width>100</width>
      <heigth>100</heigth>
    </size>
    <location>
      <x>102</x>
      <y>220</y>
    </location>
    <id>1</id>
    <numberOfInputPins>1</numberOfInputPins>
    <logFile/>
    <index/>
    <parametersFileName>D:\Software\TOUGH2\RESUS-Example - Kopie\test1.H5</parametersFileName>
  </IOProvider>
</simulation>
```

9.3.2 Template (EINGABE.tmp)

```
TOUGH2/EOS9nT MEMORY ALLOCATION
1000000   209790      8          ! MNEL,MNCON,No_CEN
05        0030                 ! MNOGN,MGTAB
20                             ! NROKMX
2                              ! ntrcmx
FLAC3D-TOUGH2-Kopplung

ROCKS----|---rhoK--|----n----|----K1---|----K2---|----K3---|--lambda-|----c----|
TON      22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
SCHAC    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
BENTO    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
BEHAE    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
KAMME    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
QUER     22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
QUERV    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
```

```
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
RICHT    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
SCHAV    22.7273E+031.2000E-011.0000E-201.0000E-201.0000E-203.2000E+009.2000E+02
        0.        0.3.2000E+00        1.        0.        0.      0.
    9     5.0000E-010.0000E+005.0000E-015.0000E-015.0000E-01
    9     5.0000E-010.0000E+005.0000E-015.0000E+001.0000E+02
BENTN    22.4654E+033.3075E-013.4467E-193.4467E-193.4467E-191.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.        0.      0.
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
BEHAN    22.3652E+033.0237E-013.4467E-193.4467E-193.4467E-191.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
KAMMN    22.4654E+032.0000E-011.0000E-181.0000E-181.0000E-181.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
RICHN    22.4654E+033.3075E-013.4467E-193.4467E-193.4467E-191.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
QUEVN    22.4654E+032.0000E-011.0000E-181.0000E-181.0000E-181.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
QUERN    22.4654E+033.3075E-013.4467E-193.4467E-193.4467E-191.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
SCHAN    22.4654E+033.3075E-013.4467E-193.4467E-193.4467E-191.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
SCHVN    22.4654E+032.0000E-011.0000E-181.0000E-181.0000E-181.3000E+008.0000E+02
        0.        0.1.3000E+00        1.        0.       0.0      0.0
    3     0.0000E+000.0000E+00
    7     5.1000E-010.0000E+005.0000E-081.0000E+081.0000E+00
#HRB     1    1.E-20        1.    1.E-13    1.E-13    1.E-13        0.    1.E+20
        0.        0.        0.        1.        0.
#THRB    1    1.E-20        1.    1.E-13    1.E-13    1.E-13        .6    1.E+20
        0.        0.     .0262        1.        0.

START----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
----*----1-MOP: 123456789*123456789*1234----*----5----*----6----*----7----*----8
PARAM-t0-|---tend--|----dt---|--maxdt--|----|     ----g----|---------|---------|
 8 09999    99990000000 01000 00 10029003.1536E+05         1         1
        0.3.1536E+133.1536E+003.1536E+10          1.0000E+014.0000E+00
1.0000E-03
1.01300000000000E+05                 1.1.50000000000000E+01
              0.0e-0              0.0e-0              0.0e-0
              0.0e-0              0.0e-0              0.0e-0
              0.0e-0              0.0e-0              0.0e-0
TIMES----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
    5    5
3.1536E+099.4608E+096.3072E+103.1536E+126.3720E+12
TRACR----1----*----2----*----3----*----4----*----5----*----6----*----7----*----8
    1    5    5    steady2HOOG08      1.0e-10 0
1 1 1 9 0     5.0e02    1.0e-1DT D
Ur235SON  0  0  0  0  0  0   0<$1:10   >3.1536E+13    2.34e2   1.00e00
tracer,trtype,pctype,nmrock,idrock,ndautr,naddid,idpare,idseq,dd00,haflif,wtmol,X_Ref




RPCAP     ---------|---------|---------|---------|---------|---------|---------|
    3             0.        0.
    8



ENDCY-------------------------------------------------------------------------
```

10 Bibliography

Abramov, S., Mannan, S., & Durieux, O. (2009). Semi-active suspension system simulation using simulink. *International Journal of Engineering Systems Modelling and Simulation*, pp. 101-114.

Adams, B. M., Bohnhoff, W. J., Dalbey, K. R., Eddy, J. P., Eldred, M. S., Gay, D. M., . . . Swiler, L. P. (2015). *Dakota, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.3 Developers Manual.*

Adams, B. M., Ebeida, M. S., Eldred, M. S., Jakeman, J. D., Swiler, L. P., Adam Stephens, J., . . . Rushdi, A. (2014). *DAKOTA, a Multilevel Parellel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, and Sensitivity Analysis: Version 6.3 Uers's Manual.* United States: Department of Energy.

Airbus-EDF-IMACS-Phimeca. (2016). *Developer's guide OpenTURNS 1.7*. Retrieved from http://doc.openturns.org/openturns-latest/pdf/OpenTURNS_DevelopersGuide.pdf

Altintas, I., Berkley, C., Jaeger, E., Jones, M., Ludaescher, B., & Mock, S. (2004). Kepler: Towards a grid-enabled system for scientific workflows. *GGF10*.

Andrianov, G., Burriel, S., Cambier, S., Dutfoy, A., Dutka-Malen, I., De Rocquigny, E., . . . Mangeant, F. (2007). Open TURNS, an open source initiative to Treat Uncertainties, Risks' N Statistics in a structured industrial approach. ESREL.

Avila, R., Broed, R., & Pereira, A. (2003). Ecolego - A toolbox for radioecological risk assessment. International Atomic Energy Agency (IAEA).

Bänecke, A., Bozau, E., Brunnengräber, A., Budelmann, H., Chaudry, S., Eckhardt, A., . . . Köhler, A. (2015). *Zwischenbericht des Verbundvorhabens ENTRIA 2013-2015 (in preparation).* ENTRIA.

Baudin, M., Dutfoy, A., Iooss, B., & Popelin, A. L. (2015). Open TURNS: An industrial software for uncertainty quantification in simulation. *arXiv preprint arXiv:1501.05242*.

Bergeaud, V., & Lefebvre, V. (2010). *Salome: A software integration platform for multi-physics, pre-processing and visualisation.*

Bergeaud, V., & Tajchman, M. (2007). Application of the salome software architecture to nuclear reactor research. *spring simulation multiconference-Volume 2* (pp. 383-387). Society for Computer Simulation International.

Broed, R., & Xu, S. (2008). User's manual for ecolego toolbox and the discretization. *Technical Report*. Stockholm, Sweden: Swedish Radiation Protection Authority.

Buhmann, D. (2000). Das Programmpaket EMOS. Ein Instrumentarium zur Analyse der Langzeitsicherheit von Endlagern. *GRS-159*. Braunschweig: Gesellschaft für Anlagen-und Reaktorsicherheit (GRS) mbH.

Cacuci, D. G., Aragonés, J. M., Bestion, D., Coddington, P., Dada, L., & Chauliac, C. (2006). Nuresim: a european platform for nuclear reactor simulation. *FISA: Conference on EU Research and Training in Reactor Systems.*

CEA/DEN, EDF R&D, OPEN CASCADE. (2015). *Salome Platform Documentation*. Retrieved from SALOME architecture : http://docs.salome-platform.org/latest/gui/GUI/salome_architecture_page.html

CEA; EDF; OPENCASCADE. (2016). Retrieved from SALOME7 THE OPEN SOURCE INTEGRATION PLATFORM FOR NUMERICAL SIMULATION: http://researchers.edf.com/fichiers/fckeditor/Commun/Innovation/logiciels/salome/Plaquette_SALOME_V7.pdf

Ciecior, W., & Ghofrani, J. (2016). *Aktueller internationaler Stand der Biosphärenmodellierung im Rahmen der Langzeitsicherheitsanalyse nuklearer Endlager sowie Entwicklung und Implemtierung eines generischen Biosphärenmodells in die Simulationsumgebung ReSUS.* Institut für Endlagerforschung: www.Entria.de (in process).

Clayberg, E., & Rubel, D. (2008). *eclipse Plug-ins.* Pearson Education.

Crestaux, T., Martinez, J. M., Le Maitre, O., & Lafitte, O. (2007). *SAMO 2007*. Retrieved from Polynomial chaos expansion for uncertainties quantification and sensitivity analysis [PowerPoint slides]: http://samo2007.chem.elte.hu/lectures/Crestaux.pdf

Deville, E., Montarnal, P., Loth, L., & Chavant, C. (2009). Alliances: simulation platform for radioactive waste disposal.

Dutfoy, A., Dutka-Malen, I., Pasanisi, A., Lebrun, R., Mangeant, F., Sen Gupta, J., . . . Yalamas, T. (2009). Openturns, an open source initiative to treat uncertainties, risks' n statistics in a structured industrial approach. *41'emes Journ'ees de Statistique, SFdS*. Bordeaux.

Eldred, M. S., Bohnhoff, W. J., & Hart, W. E. (1999). Dakota, a multilevel parallel object-oriented framework for design optimization, parameter estimation, sensitivity analysis, and uncertainty quantification. *Sandia National Labs Report No. SAND99-0000*.

Facilia. (2016). *Ecolego*. Retrieved from http://ecolego.facilia.se/ecolego/show/Ecolego

Folk, M., Heber, G., Koziol, Q., Pourmal, E., & Robinson, D. (2014). An overview of the HDF5 technology suite and its applications. In *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases* (pp. 36- 47). ACM.

Ghofrani, J. (2016). *ReSUS developers manual (in preparation).* Insitut Für Endlagerforschung - TU Clausthal.

Ghofrani, J., & Li, X. (2016). *ReSUS Platform Users Handbook (in preparation).* Institut für Endlagerforschung - TU Clausthal.

Giglioli, N., & Saltelli, A. (2000). Simlab 1.1, software for sensitivity and uncertainty analysis, tool for sound modelling. *arXiv preprint cs/0011031*.

Giunta, A. A. (2002). Use of data sampling, surrogate models, and numerical optimization in engineering design. *AIAA paper, 538:2002*.

Giunta, A. A., Eldred, M. S., Trucano, T. G., J., W., & Steven, F. (2002). Optimization under uncertainty methods for computational shock physics applications. *43rd AIAAASMEASCEAHSASC Structures, Structural Dynamics, and Materials Conference*, (pp. 1642-2002).

Giunta, A. A., Eldred, M., Swiler, L., Trucano, T., & Wotjkiewicz, S. J. (2004). Perspectives on optimization under uncertainty algorithms and applications. *10th AIAAISSMO Multidisciplinary Analysis and Optimization Conference.* Albany, New York.

GoldSim Technology Group . (2016). *GoldSim: Monte Carlo Simulation Software*. Retrieved from http://www.goldsim.com/Home/

GoldSim Technology Group. (2006). *GoldSim User's Guide.* GoldSim Technology Group LLC.

GoldSim Technology Group. (2016). *The GoldSim Simulation Environment*. Retrieved from http://www.goldsim.com/Web/Products/GoldSimPro/QuickTour

Golfier, H., Lenain, R., Calvin, C., Lautard, J. J., Baudron, A. M., Fougeras, P., . . . Dutheillet, Y. (2009). Apollo3: a common project of cea, areva and edf for the development of a new deterministic multi-purpose code for core physics analysis. *International Conference on Mathematics and Computational Methods Applied to Nuclear Science and Engineering.*

Hart, W. E., & Eldred, M. S. (1998). Design and implementation of multilevel parallel optimization on the intel teraflops. *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization number AIAA-98-4707*, (pp. 44–54).

Ishigami, T., & Homma, T. (1990). An importance quantification technique in uncertainty analysis for computer models. *Uncertainty Modeling and Analysis, 1990. Proceedings., First International Symposium*, pp. 398-403.

Joint Research Center. (2016). *The European Commission's science and knowledge service* . Retrieved from SIMLAB and other software: https://ec.europa.eu/jrc/en/samo/simlab

JUnit. (2016). *JUnit*. Retrieved from http://junit.org/junit4/

Kepler Actor Reference. (2016). Retrieved from https://code.kepler-project.org

Kolditz, O., Kaiser, R., Habbar, D., Rother, T., & Thorenz, C. (2003). *ROCKFLOW-Theory and users manual, release 3.9.* Groundwater Group, Center for Applied Geosciences, University of Tübingen, and Institute of Fluid Mechanics.

Kossik, R., & Miller, I. (2009). *Goldsim user's manual.* Issaquah, WA USA: GoldSim Technology Group.

Larour, E., Schiermeier, J., Rignot, E., Seroussi, H., Morlighem, M., & Paden, J. (2012). Sensitivity analysis of pine island glacier ice flow using issm and dakota. *Journal of Geophysical Research Earth Surface (2003–2012), 117(F2)*.

Lee, M., Hwang, S., Hyung, K., & Soo, H. (2005). A dose assessment model for krs hlw repository. *Korean Nuclear Society 2005 Fall Meeting.*

Lee, Y., Hwang, Y., Kang, C., & Hahn, P. (2005). Nuclide transport calculation in near-and far-field of a reference hlw repository using amber. *Proceedings of the Waste Management.*

Lee, Y., Hwang, Y., Kang, C., & Hahn, P. (2005). Some illustrative examples from parametric studies for a reference hlw repository using acgeo. *Korean Nuclear Society 2005 Spring Meeting.*

Lee, Y., Kang, C., & Hwang, Y. (2007). Nuclide release from an hlw repository: Development of a compartment model. *Annals of Nuclear Energy, 34(10)*, (pp. 782-791).

Lee, Y.-M., & Hwang, Y. (2009). A GoldSim model for the safety assessment of an HLW repository. *Progress in Nuclear Energy 51.6*, (pp. 746-759).

Li, X. (2015). Entwicklung der Software Plattform RESUS: Repository Simulation,Uncertainty propagation and Sensitivity analysis. *PhD thesis*. Clausthal University of Technology: Institute of Disposal Research.

Little, R., Avis, J., Calder, N., Garisto, N., Gierszewski, P., Leung, H., . . . Walke, R. (2009). A preliminary postclosure safety assessment of OPG's proposed L&ILW deep geologic repository, Canada. *ASME 2009 12th International Conference on Environmental Remediation and Radioactive Waste Management.* American Society of Mechanical Engineers.

Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., . . . Zhao, Y. (2006). Scientific workflow management and the kepler system. *Concurrency and Computation:Practice and Experience 18(10)*, pp. 1039-1065.

Marrel, A., Iooss, B., Laurent, B., & Roustant, O. (2009). Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety 94(3)*, pp. 742-751.

MathWorks. (2016). MatLab.

Matsumoto, M., & Nishimura, T. (1998). Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, pp. 3-30.

Montarnal, P., Dimier, A., Deville, E., Adam, E., Gaombalet, J., Bengaouer, A., . . . Chavant, C. (2006). Coupling methodology within the software platform alliances. *arXiv preprint cs/0611127*.

Moridis, G. J., Wu, Y. S., & Pruess, K. (1999). *EOS9nT: A TOUGH2 module for the simulation of water flow and solute/colloid transport in the subsurface.* CA (US): Ernest Orlando Lawrence Berkeley National Lab.

Near surface disposal of llw. (2015). Retrieved from https://www.quintessa.org/software/AMBER/near-surface-disposal.html

Nuclear Energy Agency. (1989). *PSACOIN Level E Intercomparison. An International Code Intercomparison Exercise on a Hypothetical Safety Assessment Case Study for Radioactive Waste Disposal Systems.* Paris: OECD.

OPEN CASCADE SAS. (2016). *CEA, ANDRA & EDF - Alliances*. Retrieved from A working environment for the simulation and analysis of phenomena to be taken into account for waste storage and disposal studies: http://www.opencascade.com/content/cea-andra-edf-alliances

Open Cascade. Salome6. (2012). *the open source integration platform for numerical*.

Parkhurst, D., Kipp, K., Engesgaard, P., & Charlton, S. (2004). *PHAST--A program for simulating ground-water flow, solute transport, and multicomponent geochemical reactions.* US Department of the Interior: US Geological Survey.

Plischke, E. (2010). An effective algorithm for computing global sensitivity indices (EASI). *Reliability Engineering & System Safety*, pp. 354-360.

Plischke, E. (2016). *Computing Global Sensitivity Measures from Given Samples.* Retrieved from Clausthal University of Technology - Institute of Disposal Research: http://www.immr.tu-clausthal.de/~epl/work/GlobalSA/GlobalSensitivityIndexEstimators.html

Pröhl, G., Olyslaegers, G., Kanyar, B., Pinedo, P., Bergström, U., Mobbs, S., . . . Hallberg, U. (2005). Development and comparison of five site-specific biosphere models for safety assessment of radioactive waste disposal. *Journal of Radiological Protection*, p. 343.

Pruess, K. (1991). *TOUGH2: A general-purpose numerical simulator for multiphase fluid and heat flow.* Berekeley, California: Lawrence Berkeley Lab.

Punt, A., Smith, G., Herben, M., & Lloyd, P. (2005). Amber: A flexible modelling system for environmental assessments gms feedback. *CARDIFF 2005*, p. 250.

Qt Company. (2016). *Qt Documentation*. Retrieved from Model/View Tutorial: http://doc.qt.io/qt-4.8/modelview.html

Quintessa. (2015). *AMBER 6.0 Reference Guide.* Quintessa Limited.

Rathmann, U., & Wilgen, J. (2016). *Qwt User's Guide*. Retrieved from http://qwt.sourceforge.net/

Rozental, G. (2016). *Boost Test*. Retrieved from The Unit Test Framework: http://www.boost.org/doc/libs/1_46_0/libs/test/doc/html/utf.html

Ruark, M. D., Niemann, J. D., Greimann, B. P., & Arabi, M. (2011). Method for assessing impacts of parameter uncertainty in sediment transport modeling applications. *Journal of Hydraulic Engineering*, pp. 623 - 636.

Rubel, D., Wren, j., & Clayberg, E. (2011). *The Eclipse Graphical Editing Framework (GEF).* Addison-Wesley Professional.

Sandia Corporation . (2016). *Dakota Reference Manual Version 6.1*. Retrieved from https://dakota.sandia.gov/content/61-reference-manual

Simulation and model-based design. (2015). Retrieved from http://www.mathworks.com/products/simulink/

Tarantola, S. (2005). *Simlab 2.2 reference manual.* Ispra, Italy: Institute for Systems, Informatics and Safety, European Commission, Joint Research Center.

The kepler project. (2015). Retrieved from https://kepler-project.org/

xlslib. (2016). Retrieved from http://xlslib.sourceforge.net/

Javad Ghofrani

Berliner Straße 57. 38678, Clausthal-Zellerfeld javad.ghofrani@tu-clausthal.de

Persönliche Daten

GEBOREN AM 23.09.1985

IN IRAN, HAMEDAN/RAZAN

NATIONALITÄT IRANISCH

Berufstätigkeit

WISSENSCHAFTLICHER MITARBEITER | TECHNISCHE UNIVERSITÄT CLAUSTHAL | APRIL 2013 – MÄRZ 2016

Institut für Endlagerforschung

WISSENSCHAFTLICHE HILFSKRAFT | TECHNISCHE UNIVERSITÄT CLAUSTHAL | OKTOBER 2010 – MÄRZ 2013

Institut für Informatik und Mathematik

WISSENSCHAFTLICHE HILFSKRAFT | IRAN UNIVERSITY OF SCIENCE AND TECHNOLOGY| OKTOBER 2005 – OKTOBER 2009

Institut für Informatik

Bildung

TECHNISCHE UNIVERSITÄT CLAUSTHAL | APRIL 2013 - MAI 2016 | PROMOTION

Hauptfach: Studium der Geoumwelttechnik (Geoenvironmental Engineering). Dr. rer. nat

TECHNISCHE UNIVERSITÄT CLAUSTHAL | OKTOBER 2010 - APRIL 2013 | MASTER

Hauptfach: Studium der Informatik, M.Sc.

IRAN UNIVERSTIY OF SCIENCE AND TECHNOLGOY| OTOBER 2004 – OKTOBER 2009 | BACHELOR

Hauptfach: Studium der Software Engineering. B.Sc.

VORSTUDIUM | OKTOBER 2002 – OKTOBER 2003

Iran, Maschhad

SCHULE | OKTOBER 1991 – OKTOBER 2002

Gymnasium

Mittelschule

Grundschule