NAC-0055_R2

# Quality Assurance Project Plan

## Clive DU PA Model v1.4

11 November 2015

Prepared by

NEPTUNE AND COMPANY, INC.

1435 Garrison St, Suite 110, Lakewood, CO 80215

| | | |
|---|---|---|
| 1. Title: Quality Assurance Project Plan | | |
| 2. Filename: Neptune and Company Quality Assurance Project Plan v1.4.docx | | |
| 3. Description: QAPP for Energy*Solutions* Clive DU PA Model v1.4 | | |
| | Name | Date |
| 4. Originator | Mike Sully | 9 November 2015 |
| 5. Reviewer | Dan Levitt | 11 November 2015 |
| 6. Remarks<br><br>5 Nov 2015: Updated and converted into new format consistent with other white papers. | | |

# CONTENTS

# FIGURES

# TABLES

# 1.0 Introduction

This document describes the Quality Assurance Project Plan (QAPP) for modeling services provided for the development of a Performance Assessment (PA) model for the disposal of depleted uranium (DU) by Energy*Solutions* at the Clive, Utah facility. Throughout this document, the term Quality Assurance (QA) refers to a program for the systematic monitoring and evaluation of the various aspects of PA model development to ensure that the models and analyses are of the type and quality of that needed and expected by the client.

# 2.0 Project Management and Organization

Neptune and Company, Inc. (Neptune) has developed this QAPP for conducting work for Energy*Solutions.* This QAPP is based on the Environmental Protection Agency (EPA) QA/G-5M Guidance for Quality Assurance Project Plans for Modeling, and our company's 23-year history working in the environmental quality arena. A tiered approach is used that includes specific procedures developed by Neptune that have been developed for modeling projects. This project-specific QAPP will work as an umbrella plan that ensures quality across all tasks.

The Neptune quality program includes:

- Experienced and trained personnel who understand the QA requirements of each task.
- An experienced Project Manager.
- A corporate Quality Assurance Officer
- Task planning, tracking, and operation via internal SOPs.
- Emphasis on continuous improvement via internal reviews and customer feedback.

It is the policy of Neptune to implement a quality program designed to generate products or services that meet or exceed the expectations established by our clients. This quality policy addresses all products delivered to our Energy*Solutions* client under the contract. We will ensure quality through the use of a quality program that includes program and project management, systematic planning, work and product assessment and control along with continuous improvement to ensure that data and work products are produced of acceptable quality to support the intended use.

To achieve this goal, Neptune will assign appropriately qualified and trained staff and ensure that all products are carefully planned. Tasks will be conducted according to the QAPP or applicable SOP and any and all problems affecting quality will be brought to the immediate attention of the project or task manager for resolution. All products will be reviewed by another technical expert. Adequate budget and time will be planned to execute the quality system.

As indicated on Figure 1, the Neptune organizational structure ensures direct reporting between the Neptune Project QA Officer and the Project Manager. This structure requires that all Neptune technical staff report to the Neptune Project Manager who is responsible for the work.

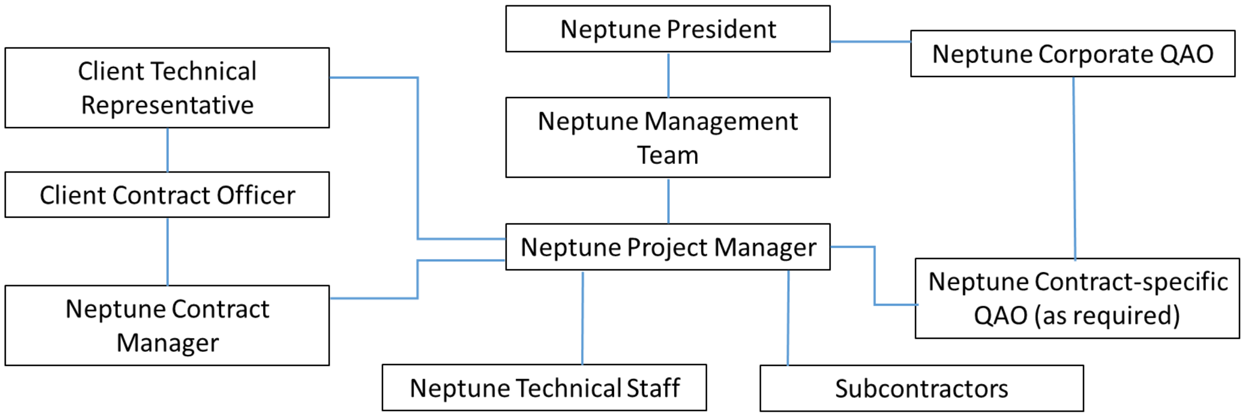


**Figure 1. Neptune Organizational Chart**

The Neptune Quality Assurance Officer has the authority and responsibility to ensure that the project-specific QAPP is implemented by Neptune staff. Roles and Responsibilities for this project are detailed in Table 1. The QA aspects of the project are handled by those project members responsible for any particular part of the project. The lead modeler is responsible for QA for the GoldSim models. For probabilistic models, the lead statistician is responsible for QA of statistical routines and products that feed into the model. The responsibility for other QA tasks may be assigned to other project members at the direction of the lead modeler or lead statistician. The *model custodian* is responsible for configuration control of the model. The role of *model custodian* may be assumed by any project team member, but only one person at a time may be the custodian.

# 3.0 Personnel Qualifications and Training

Neptune technical staff is composed of highly qualified chemists, engineers, statisticians, IT professionals, QA specialists, and biologists with advanced degrees in their fields and direct training experience. Many of the Neptune staff have participated in GoldSim training courses and GoldSim User Conferences. Qualifications for the staff are shown in Table 1. Each Neptune employee or contractor involved with this project will be required to read this QAPP and associated standard operating procedures (SOPs).

**Table 1. Roles, Responsibilities, and Training**

| Roles and Responsbilities | Personnel | Training |
|---|---|---|
| Project Manager | Paul Black | Ph.D. Statistics |
| QA Officer, Hydrologist | Mike Sully | Ph.D. Soil Science<br>GoldSim Training |
| Technical Lead | John Tauxe | Ph.D. Civil Engineer, Professional Engineer (New Mexico), GoldSim Training |
| Modeler, Geochemist | Katie Catlett | Ph.D. Soil Science<br>GoldSim Training |
| Modeler, Hydrologist | Dan Levitt | Ph.D. Soil Science<br>GoldSim Training |
| Statistician | Stephanie Fitchett | Ph.D. Statistics |
| Modeler, Hydrologist | Amy Jordon | Ph.D. Hydrology<br>GoldSim Training |
| Modeler, Statistician | Tom Stockton | Ph.D. Environmental Modeling<br>GoldSim Training |
| Modeler, Exposure and Dose Assessment | Ralph Perona | M.S. Environmental Health, DABT<br>GoldSim Training |
| Modeler, Engineer | Gregg Occhiogrosso | M.S. Environmental Engineering<br>GoldSim Training |
| Risk analyst | Robert Lee | M.S. Environmental Health |
| Modeler, Ecologist | Greg McDermott | M.S. Entomology |
| Statistician | Matt Pocernich | M.S. Environmental Engineering<br>M.S. Applied Mathematics (Statistics) |
| Statistician | Will Barnett | M.S. Ecological and Environmental Statistics |
| Technical Writer | Annette Devlin | M.A. English |

# 4.0 Project Description

Current scope under this QAPP includes four major elements: 1) DU PA model consolidation; 2) Model embankment terminology change; 3) Responses to unresolved issues in the April, 2015 Safety Evaluation Report (SER); and 4) Modification of the Class A West (CAW) evapotranspiration (ET) model to address certain SER issues. A description of the activities for each element are described below in more detail.

## 4.1 DU PA Model Consolidation

Several different models have been developed to date including: the initial DU PA v1.0; v1.2 developed in response to interrogatories; the original Deep Time Supplemental Analysis (DTSA) model; and revisions to the DTSA model that addressed changes in sedimentation rates and did not disperse the DU waste upon destruction of the mound (upon return of a lake to the Clive elevation). Consolidating these models into a single model will help respond to the SER issues more efficiently, address any future reviews more efficiently, and will bring all current models under one roof, which will be needed in the future if the PA is to be expanded to address other wastes and/or embankments.

The v1.2 model will also be updated to v11 of GoldSim – the DTSA model is already in v11. The consolidated model will need to be rerun, and results produced, including sensitivity analysis. The primary changes will be in the deep time part of the model, but the entire model will be rerun.

The new consolidated version will be labeled the Clive DU PA v1.4 model. Changes will be noted in the version change log of the GoldSim model. Supporting documentation also needs to be updated with this model revision (again, mostly for the deep time aspects of the model). This will include the v1.2 Conceptual Site Model (CSM) and Features, Events and Processes (FEPs) reports, the white papers, the parameter list document, and the final report including a revised sensitivity analysis.

## 4.2 Model Embankment Terminology Change

The terminology for the CAS Cell needs to be changed to the Federal DU Cell, and the dimensions need to be updated.

The nomenclature will be changed for the embankment from Class A South Cell to Federal DU Cell. In the model, all references to "_CAS_" will be replaced with "_FDU_". These references run throughout the model, and will require many changes to parameter names and to in-model documentation. Any change to a parameter name will require a coordinated change in the Parameters Document, and an update to QA (at least in the version change notes and Note Panes for each element changed). The terminology changes are likely to require text modifications on many elements, and nearly every dashboard and result element.

Energy*Solutions* will provide the most recent engineering drawings and the Engineering white paper will be revised accordingly. This will require including references to any new engineering drawings that Energy*Solutions* may have of the Federal DU Cell.

## 4.3 SER Unresolved Issues Responses

The scope of this work involves modifying v1.4 of the model to address the SER issues. New model versions will be created to address each of the issues. These model versions will be labeled with v1.4XXX, where the XXX is used to denote that these models are not sanctioned by Neptune, but rather, were developed in order to respond to SER issues. The SER issues will be investigated using four XXX models:

- Clive DU PA Model v1.4XXX Benson.gsm
- Clive DU PA Model v1.4XXX Benson Clay Liner.gsm
- Clive DU PA Model v1.4XXX Benson Deep Time.gsm

- Clive DU PA Model v1.4XXX Benson Erosion.gsm

Documentation will include the results of these four XXX models, and a discussion of the basis, or lack thereof, of the modifications included in the XXX models. These four models will help to investigate the following SER issues:

**a. UAC R313-25-8(2) and (3): Evapotranspiration Cover (lack of correlation between the alpha and hydraulic conductivity values, etc.)**

The Hydrus 1D model that is used as the basis for infiltration and water balance parameters in GoldSim will be modified so that the cap is naturalized. Input parameters for these infiltration models are derived from the distributions and methods described by Dr. Craig Benson in Volume 2, Appendix E, of the SER. Fifty of these parameter sets will be used as inputs to the naturalized Hydrus 1D model. The infiltration and moisture content results of these runs will be statistically abstracted to provide inputs to the modified GoldSim model. Tables showing average water balance components for the last 100 years of the Hydrus simulations will be prepared for 5 of the parameter sets with model results that span the observed range of net infiltration.

**b. UAC R313-25-8(2): Infiltration (lack of correlation between the alpha and hydraulic conductivity values, etc.)**

This issue is resolved as part of issue 4.3 a above.

**c. UAC R313-25-25: Erosion of Cover (clarification of certain issues relating to Appendix 10 to the DU PA version 1.2, June 5, 2014)**

Appendix 10 of the DU PA Model Final report will be revised to more clearly explain the SIBERIA model (v1.2). Figure 2 in Appendix 10 will be revised to include all realizations that were performed, or a new figure will be added with all the realizations for clarification (the figure currently shows the first 5 simulations, not all 1,000 that were run).

The influence of cover thinning on net infiltration will be investigated using Hydrus 1D models of the cover system.

Clarification will be provided in Appendix 10, Figure 2 that the distribution of cover area associated with a channel depth is unaffected when all the realizations are considered.

Cover thinning (erosion) will be included in Hydrus and, consequently, in the GoldSim model. The CSM document and other supporting documents will be updated to further explain the conceptual model underlying the v1.4 model. The v1.4XXX Benson Erosion model will be run, results obtained, sensitivity analysis performed, and a technical memorandum written to document the results and compare results to the v1.4 model results.

**d. UAC R313-25-25(3) and (4): Frost Damage (need to resolve concerns with assumed recurrence intervals, estimated frost penetration depths, and hydraulic property estimates)**

The SER issue indicates that Energy*Solutions* should account in modeling for substantial disruption of near-surface layers above and within the radon barriers by frost, with accompanying decreases in ET and increases for initially low-permeability soil in both hydraulic conductivity and correlated values, which could affect modeled infiltration and radon release rates. These are the types of processes accounted for by using the naturalized cover material properties for the modeling to be provided for issue 4.3 a. This issue will be addressed through the analysis to be done in 4.3 a.

**e. UAC R313-25-24(3) and (4): Effect of Biologicals on Radionuclide Transport (need to account for natural increases in cover permeability over time)**

The SER issue indicates that an increase in cover permeability will occur in response to biotic activity. These are the types of processes accounted for by using the naturalized cover material properties for the modeling to be provided for issue 4.3 a. This issue will be addressed through the analysis to be done in 4.3 a.

The response should also indicate that the maximum rooting depth currently used in the DU PA model extends below the lower radon barrier.

**f. UAC R313-25-8(2): Clay Liner (lack of increase in Ksat values over time; lack of correlation between the alpha and hydraulic conductivity values)**

These changes will be implemented in the GoldSim model in conjunction with model v1.4XXX Benson Clay Liner. There could be some small change in GoldSim model results because the saturated hydraulic conductivity (Ksat) for the clay liner affects water content in the clay liner layer, but not significantly. The model will be run and a table of results produced to show that there is no significant difference.

**g. UAC R313-25-8(10): GoldSim Quality Assurance [the relationship between the process level model (i.e., HYDRUS) abstractions and the primary model (i.e., GoldSim) results needs to be demonstrated]:**

Table 4-1 in the SER shows that the HYDRUS and GoldSim infiltration rates are different. The GoldSim and HYDRUS infiltration rates need to be compared and some investigation performed to fully address the SER issue. This might also require running the GoldSim model to provide a basis for discussion.

More generally, a discussion of scaling for PAs needs to be included up front and center in the PA documentation explaining why use of standard errors of data is more appropriate than standard deviations for parameter distributions.

**h. UAC R313-25-9(5)(a): Deep Time Analysis**

The v1.4 model will be modified, which will incorporate the latest DTSA model, as follows:

1. The material above the DU waste will be modeled as Unit 3 to account for the expected grain-size characteristics of intermediate lake sediments and an expected southern flux of long-shore drift sand from the Grayback Hills southward toward the Clive site.

2. The intermediate lake sedimentation rate will be changed to 10 times the large lake sedimentation rate.
3. The standard deviation of the eolian deposition rate will be used instead of the standard error of the mean.

This will result in a v1.4XXX Benson Deep Time version of the model. A technical memorandum will be prepared to discuss the results, show sensitivity analysis, and compare results to the current deep time model results. The results will focus on sediment and water concentrations, but will use receptor scenarios from the main model as well to provide dose estimates for comparison.

## 4.4 Class A West ET Cover Model Revisions

The ET cover model for Class A West also needs to be modified to accommodate some of the SER issue requirements. In particular, SER issues a, b, and c need to be addressed (while issues d and e will be included implicitly).

The current Hydrus 1D models applied to the ET cover will be revised to accommodate input from Dr. Craig Benson on the correlation between hydrologic input parameters, the cap will be naturalized, and some thinning of the cap will be accommodated if infiltration is found to be affected by erosion. These changes effectively cover the frost and biotic issues (d and e).

Assuming that the 12 in. ET layer cover model will be used as the basis for the SER issue modifications, the following Hydrus 1D models will be run:

1) the current based model with an updated leaf area index;
2) the current model modified to a naturalized cap, and using input values suggested in the SER issue responses; and
3) the model modified again to allow for a thinning cap, which will be run by thinning the entire cap by the same amount (if erosion is found to impact net infiltration for a naturalized cover).

Note also that the thinned cap will be implemented at time 0 – otherwise Hydrus would need to be applied to different cap thicknesses over time, which would be computationally intensive/expensive.

For each model, 50 simulations will be run, and the resulting output will be made available for revising the RESRAD models. Because the RESRAD modeling is deterministic, a reasonable deterministic statistic (e.g., the mean or upper confidence bound on the mean) will be selected for each output parameter from Hydrus that is used in RESRAD. The purpose of running many simulations is to evaluate the conditions under which the Hydrus output results change.

The results of the different models will be described in a technical memorandum report. If any other changes are needed to the current report, those will be made as well. The technical memorandum will address the changes that have been requested through the SER, and why/how these changes are counter to the conceptual model.

# 5.0 Quality Objectives and Model Performance Criteria

Systematic planning to identify required GoldSim model components will be accomplished through the development of a CSM for the disposal of depleted uranium at the Clive facility. The CSM describes the physical, chemical, and biological characteristics of the Clive facility.

The CSM encompasses everything from the inventory of disposed wastes, the migration of radionuclides contained in the waste through the engineered and natural systems, and the exposure and radiation doses to hypothetical future humans. These site characteristics are used to define variables for the quantitative PA model that is used to provide insights and understanding of the future potential human radiation doses from the disposal of DU waste. The content of the CSM provides the basis for selection of the significant regional and site-specific features, events and processes that need to be represented mathematically in the PA model. A report describing the CSM will be developed as part of Task 1.

As described in Section 4.0 the objective of the PA is to provide a tool for determining if specific performance objectives will be met for land disposal of radioactive waste set forth in Title 10 Code of Federal Regulations Part 61 (10 CFR 61) Subpart C, and promulgated by the NRC. The quality objective for the model is to provide results that are consistent with the site characteristics, the waste characteristics, and the CSM. If data are available, the demonstration of consistency will be supported by available site monitoring data and other field investigations. The model predictions of transport of radionuclides and the inadvertent intrusion into the disposal facility, and the sensitivity and uncertainty of the calculated results should be comprehensive representations of the existing knowledge of the site and the disposal facility design and operations.

# 6.0 Documentation and Records

Subversion version-control software will be used to maintain records of ownership and traceability of all project-specific files and database contents. Original data are stored in version-controlled repositories. Additions, deletions and file modifications within the repository are tracked by the version control software, which documents the file user and the date and time of modification. The version control software also offers a "compare between revisions" feature for text files that denotes line-by-line changes between previous and current versions of a file. User-entered comments are also maintained by the version control software as files are added, deleted, or modified. Version control of records is described in more detail in the Subversion SOP in Appendix A.

Internal documentation of the GoldSim model, version change notes, change log, model versioning, and model error reporting and resolution are described in the GoldSim Model Development SOP in Appendix B and the Issue Tracking SOP in Appendix C.

# 7.0 Data Acceptance Criteria

The choice of data sources depends on data availability and data application in the model. The following hierarchy outlines different types of information and their application. The information becomes increasingly site-specific and parameter uncertainty is generally reduced moving down the list.

- Physical limitations on parameter ranges, used for bounding values when no other supporting information is available. *Example: Porosity must be between 0 and 1 by definition.*
- Generic information from global databases or review literature, used for bounding values and initial estimates in the absence of site-specific information. *Example: A common value for porosity of sand is 0.3.*
- Local information from regional or national sources, used to refine the above distributions, but with little or no site-specific information. *Example: Sandy deposits in the region have been reported to have porosities in the range of 0.30 to 0.37, based on drilling reports.*
- Information elicited from experts regarding site-specific phenomena that cannot be measured. *Example: The likelihood of farming occurring on the site sometime within the next 1000 years is estimated at 50% to 90%.*
- Site-specific information gathered for other purposes. *Example: Water well drillers report the thickness of the regional aquifer to be 10 to 12 meters.*
- Site-specific modeling and studies performed for site-specific purposes. *Example: The infiltration of water through the planned engineered cap is estimated by process modeling to be between 14 and 22 cm/yr.*
- Site-specific data gathered for specific purposes in the models. Example: The density of Pogonomyrmex ant nests adjacent to the site is counted and found to be 243 nests per hectare.

The determination of data adequacy is informed by a sensitivity analysis of the model, which identifies those parameters most significant to a given model result. Such parameters are candidates for improved quality. As the model development cycle proceeds, sensitive parameters are identified, and their sources are evaluated to determine the cost/benefit of reducing their uncertainty.

# 8.0 Data Management and Software Configuration

The acquired data, developed statistical distributions and results generated by the GoldSim model and the uncertainty and sensitivity analyses will be archived in a version-control repository as described in Section 6.0 above. Configuration management for the GoldSim model is described in GoldSim Model Development SOP in Appendix B.

# 9.0 Model Assessment and Response Actions

During model development, assessments will be conducted using a graded approach with the level of testing proportional to the importance of the model feature. Assessments will consist of:

- reviews of model theory
- reviews of model algorithms
- reviews of model parameters and data
- sensitivity analysis

- uncertainty analysis
- tests of individual model modules using alternate methods of calculation such as analytic solutions or spreadsheet calculations
- reasonableness checks

Response actions including error reporting and resolution processes are described in the GoldSim SOP and the Issue Tracker SOP.

# 10.0 Model Requirements Assessment

The purpose of these assessments is to confirm that the modeling process was able to produce a model that meets project objectives. Model results will be reviewed to ensure that results are consistent with the site characteristics, the waste characteristics, and the CSM as described in Section 5.0. Model results will be assessed to determine that the requirements of Energy*Solutions* for the use of the model have been met. Any limitations on the use of the model results will be reported to the project manager and discussed with Energy*Solutions*.

# Appendix A: Subversion SOP

**Neptune and Company, Incorporated (N&C) Internal Procedure**
***Confidential***

| General Procedure: Standard Operating Procedure (SOP) | **Document No.** NAC-0003 | Revision: 0 |
|---|---|---|
| ***Document Status: Final*** | | |
| **Title:** Subversion SOP | **Author:** Warren Houghteling<br>**Revised by: N/A** | |
| **Final Approval Signatures** | **Date** | |
| **Corporate Quality Assurance Officer:**<br>Print Name: James Markwiese<br>Signature: | 13DEC11 | |
| **N&C President:**<br>Print Name: Randall Ryti<br>Signature: | 12/17/2011 | |
| | Page 1 of 10 | |

## 1. PURPOSE

Subversion is an open source version control system. Version control is the management of changes to documents, programs, and other information stored as electronic files. Neptune uses Subversion to manage work products and other project information that can be stored electronically. Subversion has three major features that support increased productivity and better Quality Assurance:

1.1 Subversion allows for the easy sharing of files in a way that all project participants have access to the latest file version, obviating the need to send emails back and forth with updates to work products.

1.2 Subversion keeps a copy of every "committed" version of the file in its database, making it easy to go back to earlier versions of a file. No file version is ever deleted in subversion.

1.3 The progression of changes in any file can be tracked via the comment feature, which allows the user to add a comment describing what had changed each time they commit an edited version of a file to the database.

## 2. SCOPE

This Standard Operating Procedure (SOP) applies to all N&C projects requiring documentation of version control for quality assurance activities.

## 3. RESPONSIBILITIES

3.1. *N&C Corporate Quality Assurance Officer (QAO)*: Maintains current record of this SOP and may modify as the developments in contracts or internal N&C procedures warrant. Conveys proposed modifications to contract-specific quality manager and program manager.

3.2. *N&C Contract Specific Quality Manager*: Recommends modifications to this SOP when appropriate and as needed to meet contract specific QA requirements, and drafts recommended changes for review.

3.3. *N&C Program (Contract) Manager*: Ensures all Technical Staff working on the contract are trained to the internal quality SOPs that pertain to the contract and that these procedures are implemented. Reviews and approves SOPs that relate to the contract. Works with Contract Specific QA Manager to execute contract specific modifications to this SOP.

3.4. N&C Technical Staff: Maintain current training on, and implement this SOP. Recommend modifications to this procedure when appropriate by discussing their ideas with the contract QAM and/or Program Manager, to maximize their effectiveness. Participates in any and all assessments related to work under the contract, to ensure the Quality Management Plan and related SOPs are routinely implemented.

## 4. DEFINITIONS

Definitions relevant to this SOP are provided in the following section.

## 5. PROCEDURE

As the Subversion online manual (http://svnbook.red-bean.com/) states, Subversion is a centralized system for sharing information. At its core is a repository, which is a central store of data. The SVN repositories live on a central server, SVN.neptuneinc.org. New repositories can be created on the server at any time. To the user, a repository appears as a collection of files and directories (although they are not actually stored that way on the SVN server).

Users access the contents of a repository by “checking out” a local copy of the repository. This process copies files from the repository to the user’s computer, creating a local “working copy” of the repository. The user can then make changes to their local copy and “commit” these changes back to the repository, so they become part of the centralized data store. To get the latest changes committed by others, the user should always “update” their repository before working on a given file. Updating pulls down any new changes from the server that are not yet part of the user’s working copy (see Section 5.4.3).

Repositories have typically been created on a per-project basis, but some have instead been created to house all the data associated with a particular client (for example, the EPA repository). The latter approach produces very large repositories, which can make downloading the whole repository time consuming, especially for users outside the Los Alamos office where the server resides. However, this can be worked around by the user checking out only the sub-folders they need from a given repository. This will be discussed in more detail later in this SOP.

### 5.1 *Accessing Repositories*

To access Neptune's subversion repositories, you will need two things:
1) a subversion user account on the server
2) a client program running on your computer which can interact with the subversion server to allow you to check out, update, and commit files

### 5.2 *Obtaining a Subversion Account*

This should be done automatically as part of your new-employee setup but any member of the IT team can also set yours up. You will receive a username and a password, both of which need to be submitted for most SVN transactions. Fortunately, all SVN clients provide the opportunity to cache your identity so that you do not have to repeatedly enter your credentials.

### 5.3 *Subversion Clients*

#### 5.3.1 *Windows GUI*

On Windows machines, the main client we use is Tortoise SVN, which is available from Tigris.org. Its home page is http://tortoisesvn.tigris.org/. Downloading, installing and periodically upgrading Tortoise SVN is a straightforward process, but IT staff will always be glad to offer assistance if needed. Tortoise works as a plugin to Windows Explorer (NOT Internet Explorer the web browser, but the file explorer); once you have Tortoise installed, you will see special icons next to files that are part of working copies, and you will have access to SVN commands via right-clicking on any file or folder in Windows Explorer.

Other clients are available – the other client that software developers use is a plugin to the Eclipse development environment called Subclipse (also from Tigris).

#### 5.3.2 *Mac GUI*

There are two main Mac clients currently in use at Neptune, SCplugin (http://scplugin.tigris.org/), which mimics some of the Tortoise functionality but unfortunately does not have all features enabled on the latest OS version (Snow Leopard), and svnX (http://www.lachoseinteractive.net/en/community/subversion/svnx/), which has a richer feature set but a very different UI concept. Both clients are useful and can even coexist on the same machine. As is the case on Windows, plugins are also available for various development environments (e.g., Netbeans, Eclipse).

#### 5.3.3 *Command Line*

On Linux and other Unix-based systems (including the Mac), there is a command-line client program called SVN. The command line client is the most flexible and powerful way to interact with subversion, and may be needed in special situations to address issues that the GUI clients cannot handle. In these cases, IT personnel can lead you through the necessary steps.

### 5.4 *Getting Started with Subversion*

Your first experience with subversion will likely involve someone on your project team telling you to check out a repository (or sub-section of a repository) so you can examine and/or modify files. You will need the URL of the repository (or sub-directory) to be able to check it out. All Neptune SVN URLs will begin with http://SVN.neptuneinc.org/repos followed by the repository name. So if I wanted to check out the entire Neptune repository (not recommended, as it is very large), I would use the URL http://SVN.neptuneinc.org/repos/neptune.

#### 5.4.1 *Trunk, Branches, and Tags*

Most repositories have three top-level directories called trunk, branches, and tags. The trunk represents the main line of work in the repository – the branches and tags folders have specialized uses, which will be discussed later (they are mainly relevant to programmers). When someone asks you to check out the "project1" repository, and that repository has a trunk, the URL you will want to use is http://SVN.neptuneinc.org/repos/project1/trunk. However, the name of the directory you will create to check the files out into should be called project1, so you will know what repository you are working with.

#### 5.4.2 *Checking Out*

Once you have been given the URL of the repository you want to check out, you will enter that URL into your subversion client as part of a "checkout" operation. Depending on your client, you may need to create the containing directory first, or the client may do it for you if you indicate a directory that does not yet exist. Either way, the files you have requested will be copied from the SVN server to the location you have specified. Subversion does NOT CARE where on your machine you chose to store your files. Subversion keeps hidden "metadata" folders inside each folder of your working copy. One of the things these metadata folders keep track of is what URL on the server the current directory corresponds to. This means that you can move the location of the working copy on your computer, and this will *not affect subversion at all* – it still knows where to go on the server to get updates for those files, or commit changes to those files.

If the repository is large, and especially if you are not in the Los Alamos office where the SVN server resides, this initial checkout could take a long time. Your client will show you a running progress display, usually listing each file that is pulled down from the server. If the listing seems to get "stuck" on a particular file, that probably means that the next file in the list is very large, as the files are not listed until their download is complete. Occasionally, you will some kind of "timeout" error message during a long checkout. In this case, it almost always works to simply update your working copy to get the rest of the files (see the next section for updating).

### 5.4.3 *Updating*

As time passes, other team members may make changes to files in the repository you have checked out. The only way for you to see these changes is to update your working copy of the repository. Your SVN client will allow you to select any directory or file in a working copy and request that it be updated. Usually, you will want to pick the top-level directory, so you can get all the updates at once. As with checking out, your client will give you a listing of files, but in this case it will only be files that have versions newer than the one you already have in your working copy. If nothing has changed, you will see a message confirming that your working copy is already at the latest version, for example "at revision 258."

#### *5.4.3.1 Conflicts*

If you have changed a file in your working copy, and someone else has changed the same file in their working copy and committed (uploaded) their change back to the server, you may get a conflict notification. If the file is plain text, and the changes in the repository are in a different part of the file than the changes you made, you will see a notification that those changes have been merged into your version of the file (there will be a G after the file name in the list of changes). However, if your text changes conflict with the changes from the repository, or if the file is a binary file, you will get a conflict. We will talk about resolving conflicts later in this document.

### 5.4.4 *Committing*

When you have made changes to one or more files and want to publish those changes back to the repository, you need to commit them. Your SVN client will allow you to select a file or directory and issue the commit command. The client will show you a list of the changed files it found, and offer you the option of unselecting any files that might have changes you are not ready to commit. It will also provide you a space to enter a comment describing the changes made to the file(s) in question. It is critical that a meaningful comment always be filled in. This requirement will be discussed in more detail later in the document.

5.4.4.1 *Why Commits Can Fail*

The main reason that a commit will fail is if one of the files to be committed is not the latest version from the repository. Subversion will not allow you to potentially overwrite someone else's changes. For example, you cannot commit a file that is based on an earlier version than the latest version from the repository. When a commit fails for this reason, the only thing to do is to update. If the file is a text file, you may find that the changes in the repository are simply merged into your file. However, the most likely scenario is that you will get a conflict, which you will then have to resolve (see Resolving Conflicts later in this document).

Practically speaking, this means that just before you begin editing a file, you need to do an update to make sure you have the latest version. Also, if the file is binary (e.g. a MS Word document), you will want to let other members of your team know that you are editing the document, so that they won't start editing in parallel. Of course, for large documents, there are strategies that allow for editing files in parallel when you know that your changes will not conflict with your colleagues' (for example when two people are editing different sections of the document). These strategies will be discussed later under the Workflow section.

5.4.4.2 *Reverting Changes*

Sometimes you may be working on a file and wish to discard all your changes an return to the base revision from the repository. This might happen if you were to realize that you had been modifying the wrong file, or for a variety of other reasons. The revert command will discard all local changes and restore your working copy with a "pristine" version of the last version of the file or files you checked out.

Sometimes reverting is the best way to resolve a conflict. You can always save your version of the changed file to a different location and then revert the conflicted file. This will give you the latest file from the repository, and allow you to examine that file and see how it differs from yours, so you can incorporate your changes into the new version.

5.4.5 *Adding New Files or Folders*

Generally, there are two kinds of new files we add to a repository. The first are new Neptune-created files, which may become work products or simply supporting project information. In these cases, it is REQUIRED to enter a comment describing the purpose of the file and perhaps its initial content.

The second type of files we add to repositories are files received from outside sources – reports, data, communications from clients, meeting minutes, etc. In these cases it is CRUCIAL that the comment contain as much detail as possible about the provenance of the file. Being able to track down exactly where we got the file and from whom is crucial to the QA process. So the comment “adding new Eco data” is fairly useless, whereas “adding new mammal field data received from Brett Tiller via email on 7/21/2008” gives us solid backward traceability to the source of the data.

If you create a new file or folder inside a directory that is part of your working copy, it has no effect on the repository until you first add the file to the working copy and then commit that addition. Most GUI clients allow you to combine these operations by including new files in the list of changes when you begin the process of committing a directory. New files will usually appear with a question mark next to them. If you check the box next to a new file, you are telling the client program to first add the file to the containing directory and then include that addition in the final commit operation. Some GUIs will have a check box that allows you to toggle whether or not new files are shown in the commit list.

### 5.6 *Subversion Workflow*

This section describes the workflow process involved in using Subversion.

#### 5.6.1 *Repository Creation*

A repository can be created at any time by a member of the IT staff. Repository names must conform to the following requirements (not that not all existing repositories conform):

- all lower case
- no spaces – use underscores instead
- alphanumeric characters only – no special characters

Repositories are created on an as-needed basis. Once again, communication is key – team members should decide if their project needs a new repository or if it best fits inside an existing repository.

The structure of the files within the repository is also a team decision. Several templates have been used on different types of projects. Specific template examples may be made available in the future to use as starting points for new projects.

#### 5.6.2 *Working with Existing Repositories*

You always have the option to check out an entire repository, or just a subsection of a repository. The only difference between the two is the URL that is passed to the checkout command. To check out an entire repository, your URL will look like this:

http://SVN.neptuneinc.org/repos/repository_name/trunk
or, in the case of a repository with no trunk,
http://SVN.neptuneinc.org/repos/repository_name
If you only want to check out a sub-section of the repository, you simply include the path to the sub-section in your URL. Here is an example of how to check out just the QA folder (containing the new company QA plan documents) from the Neptune repository:
http://SVN.neptuneinc.org/repos/neptune/trunk/QA
This way you only get a folder with three documents rater than an entire repository.

5.6.3 *Repository Browsing*

Many of the GUI clients include a feature that allows you to "browse" the repository on the server. By entering the base URL of the repository (for example, http://SVN.neptuneinc.org/repos/neptune) in the browser window, you can view the structure of the repository as it is on the server without having to download anything. This is a great way to figure out what you might need to check out for a given purpose. For example, the browser will show you that under the trunk of the Neptune repository there is a Business Development folder, which in turn contains a proposals folder. If you are just interested in seeing the proposal work done for DOD, you can just check out the DOD folder from inside the proposals folder. Most repository browser GUIs allow you to select a sub-folder from within a repository and ask to check it out. At worst, you can use the browser view to see how to build the URL you will need to check out the sub-folder you are interested in.

One thing that a repository browser GUI will NOT do is allow you to see all the different repositories on the server. To see a list of all repositories, visit to the password-protected web page at http://repositories.neptuneinc.org/index.php. You can get the username and password from one of the IT staff.

*5.6.4 Making Changes*

There are three kinds of changes you can make to a repository:
1) Modify existing files in a repository
2) Add new files to a repository
3) Reorganize the structure of a repository

5.6.4.1 *Modifying Existing Files*

As noted earlier (Section 5.4.1), to make sure that you are working on the latest versions, always do an update before you begin modifying files. Also, especially in the case of binary files, notify other team members that you will be modifying the file(s).

*5.6.4.2 Using Locks to Enforce Serial Editing of Binary Documents*

The best way to avoid conflicts when editing files is to use subversion's locking feature. Both svnX on the Mac and Tortoise on Windows give you access to this feature. Locking a file is simple. First be sure you have the latest version of the file by running an update. Then use the GUI (or command line) to invoke the lock command (you will get an error message if a more recent version of the file exists in the repository). Once a file is locked, no one else can commit changes to that file – they will receive an error when trying to commit, telling them the file is locked and the name of the user who has the lock.

Therefore, when editing a binary file, one should ALWAYS lock the file first. If someone else already has the file locked, you will get an error with the lock owner's username, and you know that you need to wait for that team member to finish his or her edits before you can work on the file. If you successfully gain the lock, you can be sure that no one will commit a new version that will then cause a conflict when you try to commit yours. When you commit your version of the file, the lock is automatically released.

In case someone locks a file and then forgets about it and goes on vacation, locks can be broken (you may need help from an IT staff member to do this). Locks are not a strict enforcement mechanism – rather they are a way to enhance team communication.

*5.6.4.3 Editing Binary Documents in Parallel*

In cases of large binary documents with many sections, team members may work on a file in parallel, with the understanding that the different team members are working on different sections of the file. When one team member is ready to commit their changes, they may do so, and the other member(s) then need to update their versions. Before doing so, they should save their versions with changes to a location outside of their working copy, or save their changes to a new filename, perhaps with their initials appended (for example, save Report1.docx as Report1_WH.docx). This way, before the other members update, they can revert their changes in the repository to avoid a conflict when they updated to get their colleague's changes (the revert operation can also happen after the conflict – this will discard all local changes and leave the working copy with the latest version from the repository). The next team member to finish their edits can then copy just their section into the new version of the document and commit those changes. As discussed in the previous section, locks can be used to enforce the order in which changes are made to the document. Needless to say, this process requires good communication among team members to make sure that no ones changes are unintentionally overwritten.

In all cases it is a REQUIREMENT of N&C QA policy that a comment be entered summarizing the changes to the file as part of the commit process. This is essential to leveraging the full power of Subversion to provide support for Quality Assurance by providing a clear trail of comments explaining how documents evolve over time. If the project is using Bugzilla to track tasks, the comment should include references to Bugzilla task numbers where appropriate (for more details see the Bugzilla SOP; NAC-0004_R0).

*5.6.4.4 Reorganizing the Structure of a Repository*

This operation is the one most likely to lead to confusion and errors if it is done incorrectly. As mentioned earlier in the document, each directory in a working copy keeps hidden metadata about how it corresponds to the data in the repository on the server. This means that moving directories around on your computer has NO EFFECT on the structure of the repository on the server. You must move a special "SVN move" command to let the working copy know that you want to modify the directories in the working copy by adding or removing files from the (a move operation will delete files from one directory and add them to another). The actual effect on the repository will not take place until you commit your changes which include the moved files.

Similarly, deleting files from your working copy will have NO EFFECT on those files in the repository. You must use a special "SVN delete" command to let the directory containing those files that they are scheduled for deletion. The actual deletion of the files will not take place until you commit your changes that include the SVN deletes. It is important to realize that deleting a file does NOT delete the file from the repository. It simply deletes the file from the latest version of the repository. It is always possible to go back to earlier versions of the repository to "resurrect" deleted files.

Finally, because deleting files from your hard disk does not affect the repository, this can be a good last-ditch solution for solving SVN problems. Occasionally, the metadata in some part of a working copy may become corrupted, leading to error messages when you try to update the repository or delete files. You can always delete the directory to which the error message refers and then run an update on the containing directory to get a fresh copy of the data pulled down from the repository. Of course, if you have changed files in the problem directory or any of its sub-directories, you should first copy the changed files to a location outside your working copy before deleting the problem directory. Then once you have done the update to get a clean copy of the directory, you can copy your changed files back into their appropriate locations in the working copy, and they will once again show up as changed files that you can commit.

**Receipt and Acknowledgement of**
**Neptune and Company, Inc.**
**Subversion SOP, Revision 0**

Please read the following statements and sign below to indicate your receipt and acknowledgement of the Neptune and Company, Inc. Subversion SOP (NAC-0003_R0).

- I have received and read a copy of the Neptune and Company, Inc. Subversion SOP.
- I understand that my signature below indicates that I have read, understand, and will adhere to the Neptune and Company, Inc. Subversion SOP.

______________________________

Employee's Printed Name

______________________________ _______________

Employee's Signature Date

# Appendix B: GoldSim Model Development SOP

**Neptune and Company (Neptune) Internal Procedure**

| **General Procedure:**<br>Standard Operating Procedure (SOP)<br><br>**Contract Specific:**<br>Internal Neptune product | **Document No.**<br>NAC_0040_R1 | **Revision:**<br>1 |
|---|---|---|
| ***Document Status: Final*** | | |
| **Title:**<br>GoldSim Model Development SOP | **Author:**<br>John Tauxe<br><br>**Revised by:**<br>M. Sully | |
| **Final Approval Signatures** | **Date** | |
| **Corporate Quality Assurance Officer:**<br><br>**Printed Name:** Warren Houghteling<br><br>**Signature:** | | |
| **Effective Date:** 1 Jan 2015 | | |

**Document date Stamp:** 02 Feb 2015

# CONTENTS

# FIGURES

# 1.0 Introduction

This standard operating procedure (SOP) describes the development of GoldSim-based computer models. These models are used to perform contaminant transport and dose assessment calculations as the computational basis for radiological Performance Assessments (PA). They are developed using the GoldSim systems analysis software, developed by the GoldSim Technology Group (GTG), as a principal platform, commonly in conjunction with various supporting computer programs and data sources. Throughout this document, the term Quality Assurance (QA) refers to a program for the systematic monitoring and evaluation of the various aspects of GoldSim model development to ensure that standards of quality are being met.

# 2.0 Modeling Lifecycle

GoldSim model development follows a structured process or lifecycle that requires a graded approach to QA at each phase. The lifecycle for GoldSim model development is described below and correlates with the work process shown in Figure 1. Model documentation is associated with each step of the work process.

## 2.1 Model Objectives and Context

The regulatory modeling process is seen by the National Research Council (NRC, 2007) as beginning when "…decision makers, model developers, and other analysts must consider regulatory needs and whether modeling could contribute to the regulatory process." With consensus on the value of developing a model the next step is to specify the objectives and context of the GoldSim model.

Defining the objectives of the model includes establishing who will use the model, what decisions the model will be designed to support and what model calculations are required to support these decisions.

Model context includes components such as the following (NRC, 2007):

- Determination of spatial and temporal scales,
- Determination of the appropriate level of detail for process representation,
- Identification of the proposed users, their expertise, and any constraints,
- Determination of sources and required quality of input data,
- Determination of sources and required quality of data for model evaluation,
- Definition of the inputs and outputs needed and whether they will be deterministic or probabilistic,
- Determination of the level of reliability required, and
- Determination of appropriate evaluation criteria required to demonstrate that the model is sufficiently accurate for its intended use.

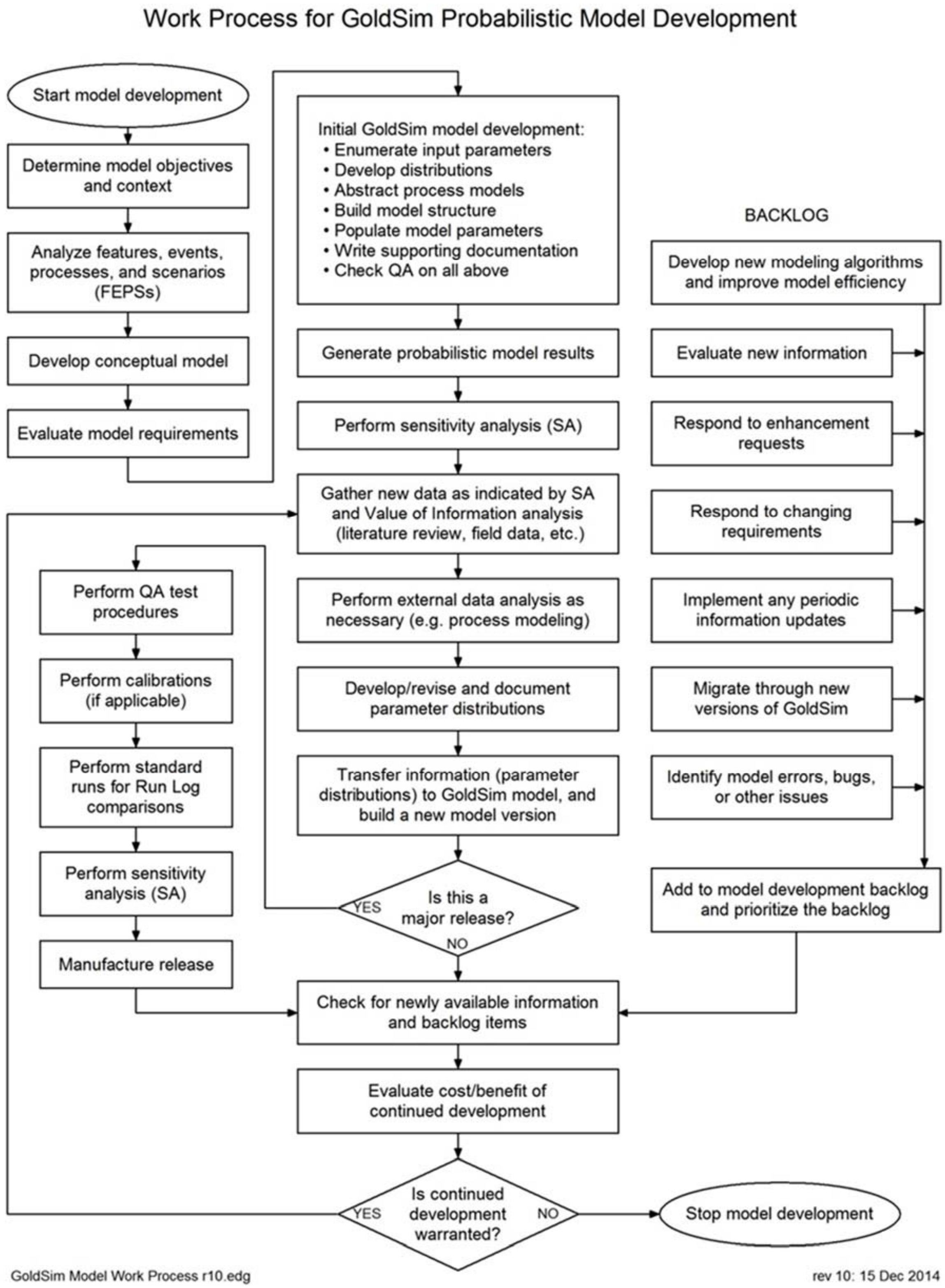


**Figure 1. Model development work process flow diagram**

Model objectives and context are documented in the requirements document described in Section 2.3.

## 2.2 Conceptual Model Development

Model development continues with the development of a Conceptual Site Model (CSM). The CSM identifies important features and processes of the system being modeled that are consistent with the existing data. Development of the CSM along with the model objectives and context form the basis for the GoldSim model design.

The CSM is documented in a Conceptual Site Model document, which explains and provides justification for the mathematical approaches for modeling geological, hydrogeological, contaminant fate and transport, demographic, and other component processes of the overall model. Existing data and literature and expert opinion are used to support the modeling approach described by the CSM.

## 2.3 Model Requirements Evaluation

The CSM provides information to determine the attributes and capabilities of the software required to meet the project objectives. These requirements and those determined in the definition of the “model objectives and context” step are compiled in a model requirements document. Model requirements also include consideration of modeling objectives to determine the reliability, certainty, and accuracy needed in predicting the performance measures for the decision process. This evaluation also includes a review conducted to verify that the GoldSim modeling platform is capable of providing these required attributes and capabilities. The model requirements document is archived in the project repository.

## 2.4 Verification of Software Installation

The GoldSim software is installed and registered as described in the GoldSim User's Guide (GTG 2010a *et seq.*). Following the installation and registration the user runs the example model “FirstModel.gsm” located in the “General Examples” directory and verifies that the output obtained matches the chart shown on page 26 of the User's Guide (GTG 2010a *et seq*.).

The GoldSim User's Guide (GTG 2010a *et seq.*) and the GoldSim Contaminant Transport Module User's Guide (GTG 2010b *et seq.*) provide complete descriptions of the features and capabilities of GoldSim and the Contaminant Transport Module.

## 2.5 GoldSim Model Development

During model development individual modelers work in parallel to model specific sub processes described in the CSM. For example, existing mathematical models are translated into specific algorithms to be used in the modeling process. GoldSim offers a level of model structure that can closely resemble a conceptual model, so the structural implementation of the GoldSim model will follow the CSM developed by the project team. As the different components of the model are developed in GoldSim, they are integrated to form a coherent representation of the overall process being studied. GoldSim's object-oriented structure facilitates this process, often allowing independently developed sub-modules to be copied and pasted into the main model. GoldSim's "self-documenting" features allow the graphical user interface (GUI) design to incorporate documentation of modeling concepts and parameter derivation, so that it is relatively easy to crosswalk between individual GoldSim pages and sections of the CSM document.

## 2.6 Model Data Inputs

### 2.6.1 Input Data Selection

The development of appropriate definitions of input parameters is guided by model sensitivity analyses, which identify those parameters most important in determining the model results. In some cases, the definition of an input value matters little to the results and in these cases less effort is expended in developing distributions. Sensitive parameters, however, warrant a closer investigation, and their input distributions are devised with great care where possible. All parameters in the model are based on some sort of information source, be it a "literature value," the result of a site-specific data collection campaign, or the result of expert professional judgment.

### 2.6.2 Input Data Placeholders

On occasion, a modeling element must be added to the model in order to proceed with construction, but no value has yet been developed. In this case, an ad hoc placeholder value is chosen so that model development may continue, and the parameter is noted as a placeholder. Before the model can be relied upon for any purpose, however, all such placeholder values must be replaced with suitably-derived and documented values.

### 2.6.3 Data Acceptance Criteria

The sources of input data for the model are various, and the quality of the source is a compromise between model sensitivity (identifying the need for high-quality data), availability, appropriateness, and the ability (budget and/or practicality) to generate data of sufficient quality. Input parameters that have a strong influence on the model results as determined by sensitivity analyses are given higher priority than those with little influence.

The choice of data sources depends on the availability and application of the data in the model. The following hierarchy outlines different types of information and their application. The information becomes increasingly site-specific and parameter uncertainty is generally reduced moving down the list.

- Physical limitations on parameter ranges, used for bounding values when no other supporting information is available. *Example: Porosity must be between 0 and 1 by definition.*

- Generic information from global databases or review literature, used for bounding values and initial estimates in the absence of site-specific information. *Example: A generic value for porosity of sand is 0.3.*
- Local information from regional or national sources, used to refine the above distributions, but with little or no site-specific information. *Example: Sandy deposits in the region have been reported to have porosities in the range of 0.30 to 0.37, based on drilling logs or reports.*
- Information elicited from experts regarding site-specific phenomena that cannot be measured. *Example: The likelihood of farming occurring on the site at some time within the next 1000 years is estimated at 50% to 90%.*
- Site-specific information gathered for other purposes. *Example: Water well drillers report the thickness of the regional aquifer to be 10 to 12 meters.*
- Site-specific modeling and studies performed for site-specific purposes. *Example: The infiltration of water through the planned engineered cap is estimated by process modeling to be between 14 and 22 cm/yr.*
- Site-specific data gathered for specific purposes in the models. *Example: The density of* Pogonomyrmex *ant nests adjacent to the site is counted, and found to be 43 nests per hectare.*

The determination of data adequacy is informed by a sensitivity analysis of the model, which identifies those parameters most significant to a given model result. Such parameters are candidates for additional measurements or more deliberate estimation. As the model development cycle proceeds, sensitive parameters are identified and their sources are evaluated in order to determine the cost/benefit of reducing their uncertainty.

### 2.6.4 Records of Parameter Values

One limitation of the GoldSim platform is that there is no straightforward way to examine all the values of inputs (data and stochastic elements) in one place. The user must search the model and open (or “mouse-over”) each input element individually in order to see its value. In order to overcome this inconvenience, all the parameter inputs are stored external to the model, in the Parameter List document.

### 2.6.5 The Parameter List

The Parameter List is a complete list of the input parameters for the model, and may consist of a text document, a workbook of spreadsheets, a database, or a combination of these, depending on the changing capabilities of the GoldSim modeling platform. Each parameter is listed in only one place, so that there is no ambiguity about the proper value of a parameter. Accompanying the listing of the parameter value in the Parameter List is a traceable reference to its origin, which may be in a White Paper or literature reference. Any change to a parameter is made to the Parameter List first, and then the change is made to the model: The value in the Parameter List is cross-checked to its source via a check print (see below), and the value in the model is then changed, noted in the Version Change Note for the modified element, and in the Change Log.

### 2.6.6 Check Prints

Whenever information (e.g. a parameter distribution) is transferred from one record to another (e.g. from a White Paper to the Parameter List) a QA Check Print process is invoked. This process is intended to positively and unambiguously document the source of information for each model input parameter or distribution.

The flow of information is from primary sources (field data, literature, expert elicitations, etc.) to White Papers that develop the input distributions (this step may not apply to all cases, and may have supplemental calculation sheets), to the Parameter List to the GoldSim model. QA check prints are maintained in all but the final step—transferring input values to the model. The check print process consists of obtaining paper copies of the data source and its destination, such as a paper from the literature and the Parameter List, for example. A comment field in the Parameter List (either a column in a table, a comment attached to a spreadsheet cell, or other location unambiguously associated with the data) identifies the value's origin. A paper copy of that page or pages of the Parameter List is attached to a paper copy of the data source (which may be simply the page from the identified source), and the QA reviewer annotates each page. Typically, a yellow highlighter is used to indicate each positively-checked value, and a red pen identifies any value that does not match. After checking each value against its source, the check print is documented with the date and the signature of the checker. Errors discovered in the process are noted, the errors are corrected in the destination document, and the values are rechecked with a subsequent check print, which is attached to the original. This process is repeated until the check prints can document that information transfers are error-free. Check prints are stored as hard copy at Neptune.

The final step of information transfer—from the Parameter Document to the GoldSim model—does not lend itself to paper check printing. However, traceability of parameter information can be maintained using GoldSim's internal QA tools, such as Note Panes and Version Change Notes discussed in Sections 3.0 and 4.0.

## 2.7 Model Evaluation

Evaluation of the proper operation of the Model is done on two levels. The overall model, as represented in the results, is subjected to benchmarking with process model results if a process model is available, and is compared to previous versions of the Model to assure that incremental changes are in line with those expected from modifications to the Model. On a submodel scale, particular parts of the Model may be evaluated independently. Many computer models that attempt to predict the outcomes of processes and events can be validated (verified) with measurable results (e.g. environmental media concentrations). Due to the nature of performance assessments, which attempt to estimate concentrations and fluxes of materials in environmental media and the potential doses or risks resulting from exposure to those materials far into the future, the results are not amenable to this type of validation. It is not possible to "test" the model at a system level to see if it has done a good job of predicting the dose to a hypothetical individual 10,000 years from now. GoldSim model evaluation includes elements recommended by the NRC (2007) that provide the evidence used to demonstrate that the models are sufficiently accurate for their intended use. These model evaluation methods used by Neptune are described in the following sections.

### 2.7.1 Scientific Basis

Documentation of the theory represented in the models in the topical White Papers demonstrates the sound scientific basis for the models.

### 2.7.2 Computational Infrastructure

Verification of the computer program GoldSim is provided in the GoldSim Verification Plan (GTG, 2010c *et seq.*). The definition of verification in the Plan is taken from Sandia National Laboratory (1995): "The process of demonstrating that a computer software program performs its numerical and logical operations correctly."

### 2.7.3 Assumptions and Limitations

Detailed descriptions of the assumptions and limitations of the models necessary for model evaluation are documented in the model or in the associated White Papers.

### 2.7.4 Peer Review

Peer review described more fully in Section 2.8 below is a documented review of the model and its application to determine if the model is sufficiently accurate for its intended use, properly documented, and meets specified quality assurance requirements.

### 2.7.5 Quality Assurance and Quality Control

The model development process is conducted under a documented QA plan that includes training and assessment of implementation of QA processes through internal reviews by staff not associated with the project.

### 2.7.6 Data Availability and Quality

The availability of data and the quality of data described in more detail in Section 2.6 are important elements in the model evaluation process.

### 2.7.7 Comparison with Analytical or Empirical Solutions

To evaluate the reasonableness of the results of a particular algorithm, the modeler may set up equation(s) both as an element in GoldSim and also using another tool, such as a Microsoft Excel Spreadsheet. This allows the modeler to compare results using two different calculation methods to provide a higher level of confidence that the algorithm has been implemented correctly in the GoldSim model.

### 2.7.8 Benchmarking against Other Models

Benchmarking consists of reproducing the deterministic results of the process model calculations using an established process model and GoldSim. This "benchmarking" is a fundamental high-level corroboration of the model implementation and calculations. Agreement between the two models serves to build confidence in the validity of the GoldSim model. Model benchmarking is documented using spreadsheet based test plans. Test plans include documentation of

- Prerequisite conditions
- Inputs
- Assumptions and constraints
- Software requirements
- Test descriptions
- Requirements—tests crosswalk
- Test log

The test plan includes the rationale for the plan, a review of the test results and identifies reviewers and their specific responsibilities.

### 2.7.9 Corroboration of Model Results with Observations

Field and laboratory data applicable to the model predictions are rarely available but for some long-term processes geologic data may be useful for corroboration.

### 2.7.10 Sensitivity Analyses

A sensitivity analysis (SA) is conducted by Neptune for system models such as performance assessments and radiological risk assessment models constructed in GoldSim. The goal of the SA is to determine which explanatory variables have the largest impact on specific endpoints of interest. That is, the SA provides the overall contribution of each input parameter to the model output. A global approach is used where all input parameters are essentially varied simultaneously. SA results are used for model evaluation leading to better understanding of model constructs and modifications to the model structure. In addition, if there is an unacceptable level of uncertainty associated with an endpoint of interest (for decision making purposes), the sensitive parameters can be targeted for effective uncertainty reduction; that is, further data or information should be collected to reduce the uncertainty on these sensitive input parameters.

### 2.7.11 Reasonableness Checking

A model can incorporate several tools for checking the reasonableness of certain inputs and results. Examples follow:

- Intermediate results are provided where they are useful for checking calculations.

- Mass balance checks demonstrate that the mass of materials (soil, water, air) and radionuclides is preserved. This is a fundamental requirement of physical environmental models (GTG 2010c *et seq.*).

## 2.8 Model Review

Model development is subject to review by a modeler different from the one who did the original model building. As parts of the model are revised, with changes in parameters, expressions, or other functional elements, or model structure, these changes are reviewed for accuracy and completeness. Any accompanying text on the model pages is also reviewed for clarity and accuracy. The modeler making the changes identifies which parts of the model are subject to review, and another Neptune GoldSim modeler examines these in detail, providing review comments to the originating modeler. The entire model is subjected to review before release to the client (see Section 4.7). This review is documented and archived in the project repository.

# 3.0 Model Documentation

## 3.1 Documentation Components

GoldSim models are documented both internally and externally. Internal documentation includes the Change Log, Version Change Notes, modeling element Note Panes, and GoldSim's internal versioning capability. External documentation includes White Papers, check prints, and a Parameter List. White Papers document the development of specific algorithms and other inputs to the GoldSim model and are intended to explain and justify the approach taken.

The scope of a given White Paper is normally concentrated on a specific topic such as engineering or processes within a subject matter area such as biotic transport. The White Paper contains more detailed discussion of literature and available data describing mathematical representations of the CSM in the model. The White Paper is used to document the source of the data used for model parameters and the methods used for developing statistical distributions for parameters. White Papers are also used to document the model used for a parameter or a process in the model if it is not a part of the GoldSim platform and explain how the process is implemented in the GoldSim model.

A typical page in the model consists of model elements and explanatory text. Illustrations such as drawings or photographs may also be used. Each page represents a modeling concept, and the model is logically divided into parts that will fit onto pages. Text at the top of each page explains the function of the page, and text juxtaposed with the model elements explains the function of the element. Each element also has a description field that is used for a short descriptive identifier. Additional details may be provided in each element's Note Pane.

The influence of one model element on another can be easily traced through the model using the "Show All Links" function attached to the triangle-shaped arrows on each side of the element graphic. The left triangle, pointing into the element, shows the other elements referenced by the current one, and the right triangle, pointing out of the element, shows the other elements that are dependent on it. By following these links, the complete interdependency of elements can be traced through the model.

## 3.2 Model Element Note Panes

Associated with each GoldSim modeling element is the optional Note Pane feature. If an element has a note, it is identified by an underlined element name. Note panes have a dual purpose in the model. They are used for general information, describing the purpose of a container or element. They also serve the QA process, as a convenient place to make notes about the source of information or the status of QA review. While most of the note pane is free-format, the QA related notes are to include a date (which can be cross-indexed to a version number using the Change Log, described below), the name or initials of the person making the note, and a description about the nature of the QA check. For example, a QA note for an entire container might read:

| | | |
|---|---|---|
| 1 Apr 05 | JT | QA for this container completed |
| 13 Apr 05 | KC | QA updated with cross-check of water tortuosity exponent parameter values |

and one for an individual element:

| | | |
|---|---|---|
| 13 Aug 04 | KC | Verified source of these data: Each value was checked against the 15th edition of the Chart of the Nuclides (General Electric Co. and Knolls Atomic Power Laboratory, 1996), wall chart version. |
| 28 Sep 04 | JT | Updated and verified source of these data: Each value was checked against the 16th edition of the Chart of the Nuclides (General Electric Co. and Knolls Atomic Power Laboratory, 1996), booklet version. |

If an element is actually changed in the process of a QA review (or for any other reason), such change is noted in the Version Change Note associated with that element. This is part of GoldSim's internal QA process, and the text of all Version Change Notes is kept in an internal database of changes, indexed to model versions.

# 4.0 Model Configuration Management

Managing the model configuration through its various versions is critical to the production of a usable modeling product that meets client requirements. The following sections discuss various topics relevant to model modification and control.

## 4.1 Model Custody

During model development, the baseline model is tracked by the lead modeler. In the event that another modeler needs to have custody of the model for development purposes, the custody will be passed to that modeler and returned when the work is finished. The current custodian is always known, and is recorded on the topmost page of the model (except in released versions). Modelers make use of the internal GoldSim versioning and Change Log in order to document changes made to the model.

A GoldSim model differs from many other software development projects in that it exists in a single binary file (with the ".gsm" extension). There are no separate files for subroutines as in a more traditional programming language like C, FORTRAN, or even Java. Therefore, the model cannot be edited by more than a single person at a time. At any given time, there is a single "main" model file. The custody of the main model must be explicitly passed from the lead modeler to another, and the custody is always known by the lead modeler, who is also the default custodian. The lead modeler may assign custody to another for a particular modeling task, but will resume custody when that task is completed. Upon return of custody, the returned model is inspected and one of two paths is chosen: 1) The returned model is maintained as the baseline model, or 2) the baseline model is modified appropriately to incorporate changes made in the returned model, and the modified baseline model is retained as the new baseline model. The baseline model resides on the custodian's computer, and is backed up by several methods, including off-site media (see Section 4.5).

### 4.1.1 Experimental Module Development

On occasion, model development requires some experimentation that may not be desirable in the main model. In such cases, a copy of the main model is made and given a unique file name in order to keep it distinct from the main model. This "branch copy" is used for module development and prototyping of modeling methods. Once the prototype of a specific module is complete, tested, and accepted, the new model parts are re-integrated into the main model, either by copying model containers and elements from the branch copy into the main model (the preferred method), or by re-entering elements directly into the main model in cases where GoldSim will not allow copying between model files. Either way, the additions and/or changes to the main model are cross checked for accuracy (by a modeler other than the one implementing the change), and the modifications are noted in the Change Log (see Section 4.2.2). At all times, however, there is only one main model file.

### 4.1.2 Criteria for Making Changes

Changes to the model occur at different levels. Minor changes to internal documentation language, including clarifications of text and correction of inconsequential typographical errors, are made as they are identified, and without formal documentation. Changes involving any type of data input or calculation that could potentially affect the modeling results are documented in all affected supporting documents (calc sheets, white papers, etc.) and in the model's Change Log.

A change to an input parameter (e.g. a distribution) may be precipitated by the following:

- QA review, in which model parameters are found to not match their values as documented outside the model. In such a case, the value and its QA records (e.g. check prints) are reviewed, and errors are corrected as appropriate.
- A decision by a subject matter expert (SME), generally in consultation with other project team members, that a value should be changed for some technical reason, such as the availability of new data on which a distribution is based. This would be considered an update, and the change would cascade through the proper sequence, from an update to the data set, through development of an updated distribution, updating of the documentation in a White Paper (if applicable) and in the Parameter List and finally an update to the model itself, with an accompanying entry in the Change Log and in the parameter element's Version Change Note (see Section 4.2.1). Each step in the change sequence is reviewed by an individual other than the person implementing the change.
- Major changes to the model, such as changing the GoldSim Species list, adding a contaminant transport process, a waste configuration, or an exposure scenario, are discussed and planned by Team SMEs.

## 4.2 Documentation of Changes

The documentation of changes made to the model is done at a level appropriate to the changes. If individual parameters are modified or added, this is documented with a note provided in the model element's Version Change Note, referencing the nature of the change, who made the change, and date of the change. The name of the changed element is noted in the Change Log, along with the model version number, date of the change, the name of the person executing the change, and the name of the reviewer of the change process. Such changes may also be noted in the element's Note Pane or that of its container.

### 4.2.1 Version Change Notes

Version Change Notes (Figure 2) are automatically attached by GoldSim to any model element that has been modified, and are used to store information about changes in any particular element. GoldSim keeps a versioning database within the Model, consisting of a list of all changes to the model between version-stamps, and the text supplied in the Version Change Notes. At any time, GoldSim can generate a report of changes made between versions. Once a model version number has been incremented, all Version Change Notes are "reset" and a new set begins for that version. Any information that is to be maintained through versions for viewing by users or reviewers, such as QA reviews, is kept in the Note Panes associated with model elements or containers. Any time an element is edited, a log entry is generated internally by GoldSim documenting the event. Note that this happens even if nothing is actually changed in the element when the "OK" button is chosen in the dialog. Use of the "Cancel" button does not signal a change.

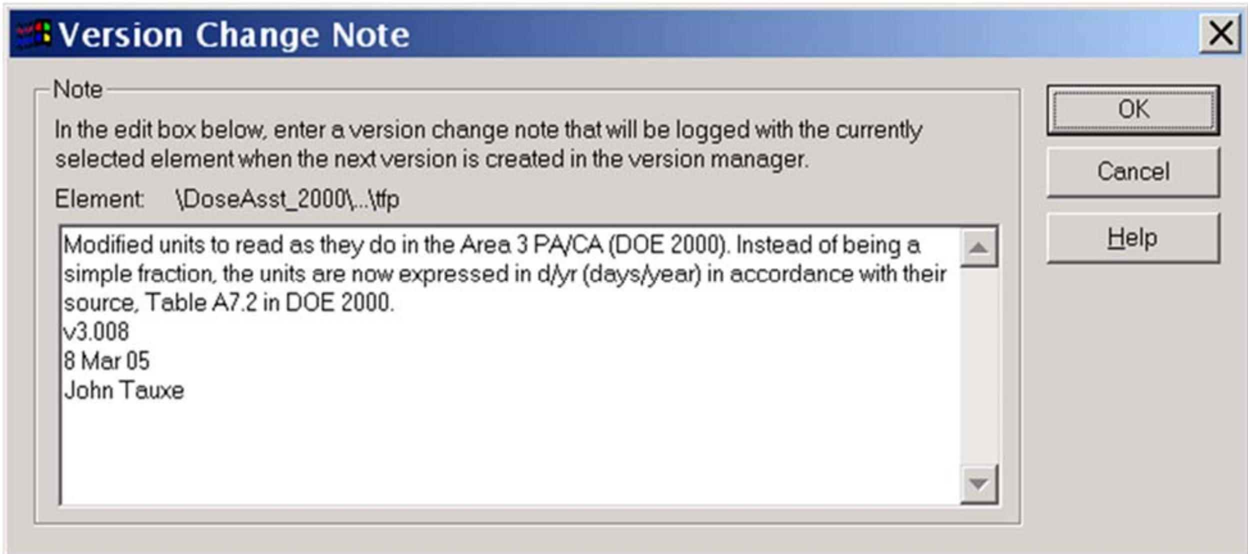


**Figure 2. GoldSim provides for annotation regarding any change in an element's definition through the Version Change Note**

### 4.2.2 The Change Log

Neptune’s GoldSim Models have a Change Log, in a block of paragraph text in the Documentation container, as shown in Figure 3. This log is maintained by the modelers, and documents when a change was made, who made it, the model version number, and descriptive details. Modifications that could potentially change modeling results are noted to the level of the element changed, with more detail included in the element’s Version Change Note or Note Pane. Modifications to explanatory text and changes to diagrams and other supporting material are noted in broad terms, such as “Modified figures depicting waste cell geometries.” Typographical corrections are generally not noted.

All of these documentation techniques are used in model development. If a change was made to the model, or if part of the model was reviewed, this will be noted in the Change Log. A note regarding the QA review (and details, if necessary) will be made in the element's note pane or in its container's note pane. The container's note pane is appropriate if there are many similar elements in the container. If a change is made to an element, either from a QA review or for another reason, GoldSim will automatically provide the element with a Version Change Note, which is used for recording the change.

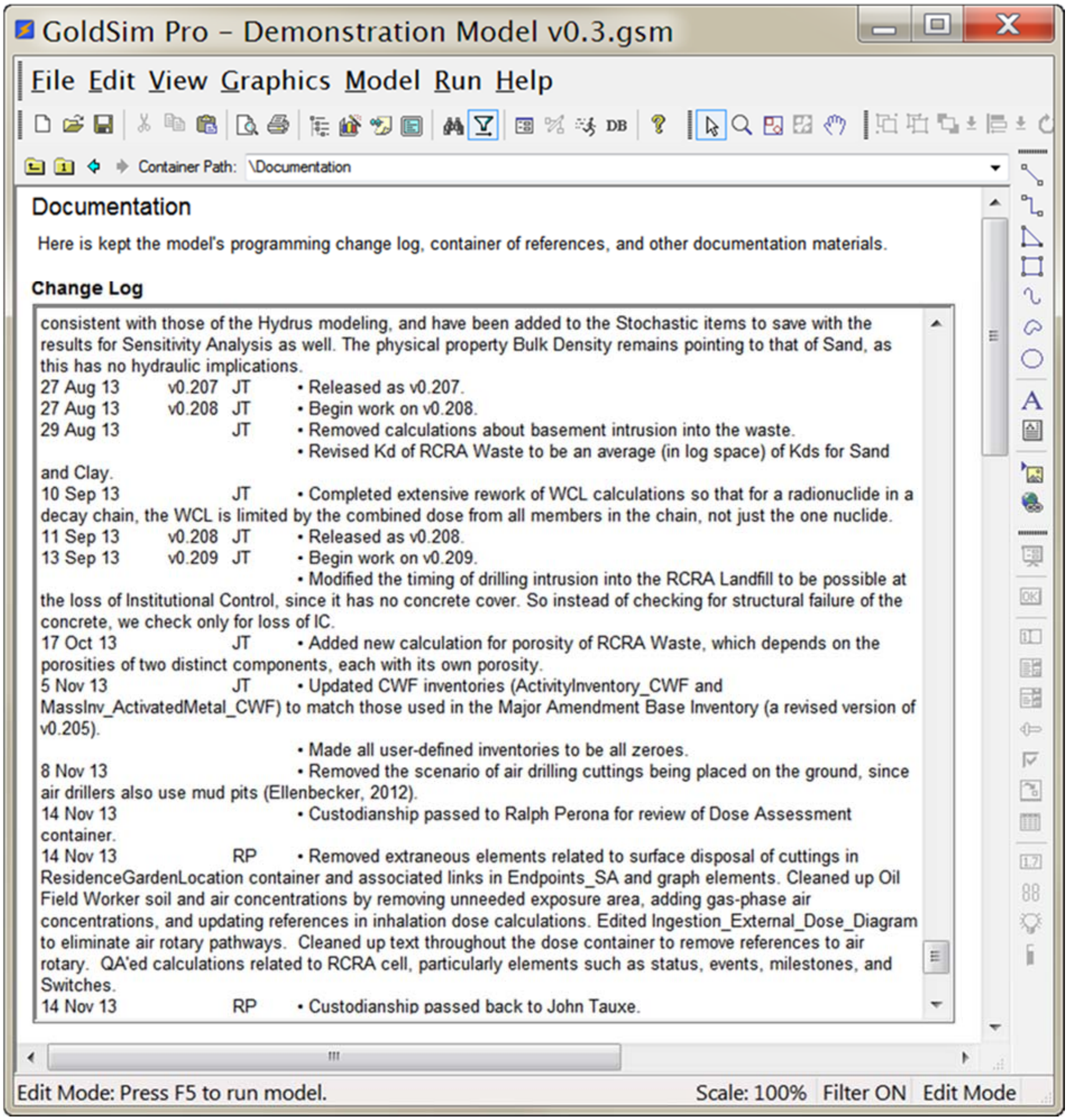


**Figure 3. The model's Change Log can be maintained using a formatted text box**

## 4.3 GoldSim Versioning

Introduced specifically as a model QA feature, GoldSim has model-level and element-level versioning built in to the Version Manager.

### 4.3.1 Model Version Numbers

At the model level, illustrated in Figure 4, version numbers are incremented at the modeler's discretion. The model version number is incremented as described below. GoldSim keeps track of changes made to the model in any given version, and can generate a report of changes made.

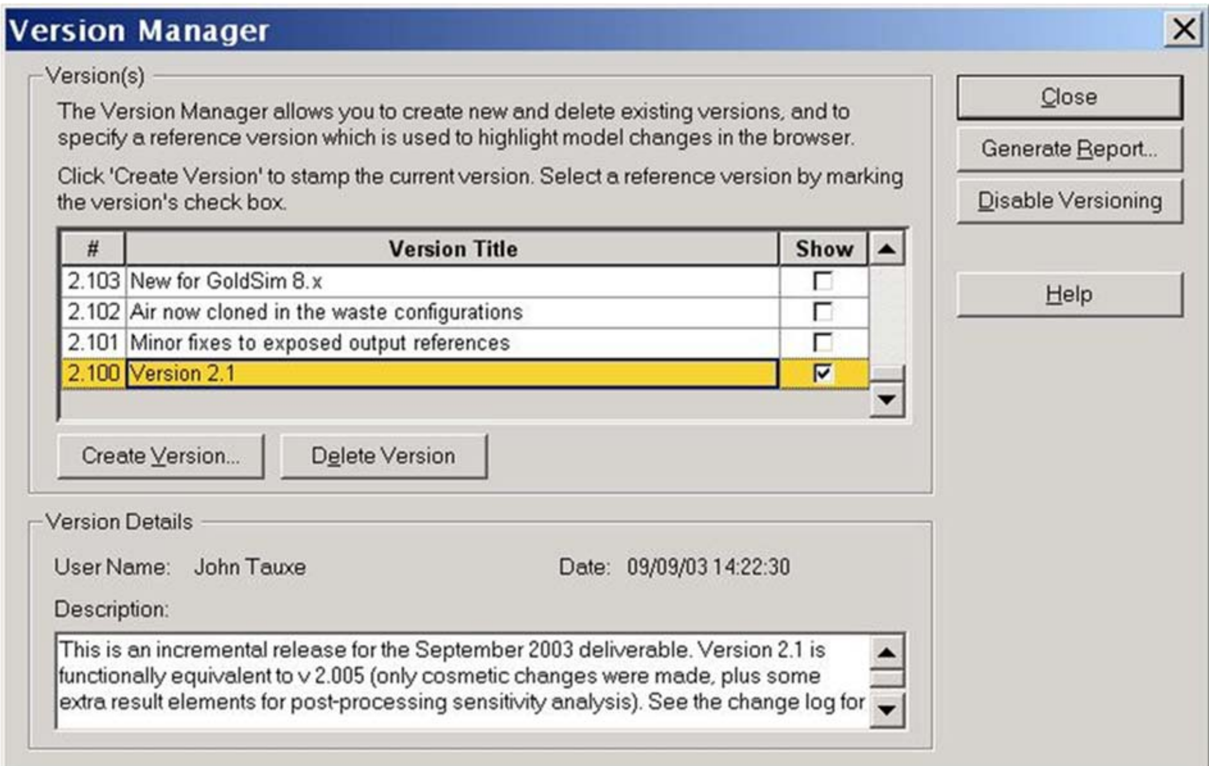


**Figure 4. GoldSim's Version Manager**

Neptune GoldSim models use versioning at two levels: Release versions and development versions. Major revisions to the model, resulting in planned releases, generally proceed in increments of X.Y, with a change in X signifying a more significant model evolution than a change in Y. The assignment of these values is subjective, and may be decided upon in coordination with the client.

Model development uses GoldSim's minor version definition, which increments the Y in the three digits following the decimal point. For example, development following the release of version 2.1 starts with version 2.101. After making some changes to the model, a modeler decides to preserve the incremental version. At this point, the version number is incremented to 2.102 and the work proceeds, with 2.101 being archived.

Day-to-day and hour-to-hour development versions are noted with letters appended to the version number, such as 2.010a, 2.010b, etc. This is done so that during the process of editing the model, any change can be easily undone. When a specific modeling task is accomplished, the model is saved with the next letter in the sequence. As the changes are tested and accepted, the letter suffixes are dropped, and these intermediate versions are generally not archived. If a problem is found during testing of daily builds, or if the model file becomes corrupted, then the modeler can easily revert to a previously saved version of the model file and rebuild the part that caused the problem. This is preferable to attempting to "undo" the work, which takes time, can be prone to error, and clutters the internal versioning record.

#### 4.3.1.1 Incrementing the Version Number

The following example illustrates the documentation of incrementing development versions, as recorded in the Change Log:

1) Make a final entry in the Change Log under version 1.034 that you are incrementing the version number:

   29 Jun 02 1.034 JJ Versioning counter updated to 1.034, and model saved.

2) Immediately change the internal versioning to 1.034 using "Model | Versioning..." (see Figure 4)

3) Save the model as "name v1.034.gsm", (any name plus the version number) overwriting all previous versions of that name.

4) Change the file attributes to "read only" so that the model file will not be inadvertently overwritten.

5) Change the front page and the Change Log entries to 1.035.

   29 Jun 02 1.035 JJ Begin work on v1.035.

6) Save the model as "name v1.035a.gsm" (or similar)

7) Begin work on version 1.035, starting at intermediate development version 1.035a.

8) After developing using intermediates 1.035a, 1.035b, 1.035c, etc., determine when to save the model as 1.035, and return to step 1) using the new version number.

#### 4.3.1.2 Creating a Versioning Report

A report can be generated from GoldSim (using the "Generate Report..." button shown in Figure 4), listing all changes to the model for a particular version. The report is a text file with global changes as well as changes to individual elements, including the text from the Version Change Notes.

## 4.4 Model Testing

Any time a change is made to the model calculations that could change the results, the effects of the change are assessed. Model testing is relatively easy using GoldSim, since the results of any element in the model can be examined through a time series or final value. This enables straightforward parallel calculations to be done in order to verify correct and consistent operation. The modeling environment also allows the simple creation of temporary elements to perform calculations parallel to any others in the model.

Model testing is most readily done on discrete parts of the model, where results of a small number of straightforward calculations can be examined. Confirmation of discrete parts of the model is done by constructing a test model in GoldSim that is focused in its analysis. Ideally, this test model is excised directly from the main model, so that all relationships and definitions are preserved. For example, to confirm that GoldSim is performing internal diffusion calculations as expected, a simple GoldSim model can be constructed to examine the diffusion of materials between various media in two cells, and the results can be compared to an analytical solution to the diffusion equation. Calculations verified in the test model give confidence in the correct operation in the model.

## 4.5 Model Backup

Preservation of electronic model files is paramount in any software development project. Several redundant methods are employed for backup of the GoldSim model files and all other files and documentation. Foremost are project files, maintained on a Neptune server, which are backed up on a separate hard drive. Incremental versions of the model are likewise backed up locally and in addition, the lead modeler keeps a copy on his/her computer, and backs that copy up to a Neptune server. Off-site backups are also maintained.

## 4.6 Error Reporting and Resolution

As errors are discovered, they must be identified, reported, and resolved. This section discusses the handling of errors in the development of a model. Formal tracking of errors, bugs, and other issues is done using an issue-tracking system maintained by the QA manager and lead modeler.

### 4.6.1 Reporting Error Candidates

Errors such as inconsequential typographical errors in supporting text are not considered in this process. Errors considered for this process include errors in parameter data entry or GoldSim programming. If an error is suspected, it is to be reported to the lead modeler along with any supporting information. It is the responsibility of the lead modeler to evaluate the error candidate and see that the issue is resolved, invoking the issue-tracking system as appropriate.

Data entry errors may be discovered in input elements (Data or Stochastic GoldSim model elements). These are also brought to the attention of the lead modeler. These or any other modeling issues are to be entered into the issue-tracking system.

### 4.6.2 Assessing Error Candidates

Once an error candidate has been brought to the attention of the lead modeler via the issue tracking system an assessment is made to determine if the candidate is in fact an error. This is usually a simple process, involving examining a mathematical expression or entered data. Real errors are subject to resolution. False errors are identified as such, noting the resolution in the issue-tracking system. If, however, the problem was due to some other cause, such as an ambiguity in documentation, the causes of the identification of a false error may require attention.

### 4.6.3 Resolving Errors

Errors, once discovered and confirmed, are usually easily remedied. Like other changes to the model, fixing an error is documented at least in Version Change Notes and the Change Log. Resolution is also noted in the issue-tracking system.

### 4.6.4 Error Resolution Verification

Checking the error resolution may be as simple as cross-checking an input value with the value in the Parameters List to ensure it is correct. Alternatively, a modification to an expression may involve an independent check of the calculation, using a spreadsheet, calculator, or a separate GoldSim model.

### 4.6.5 Error Impact Assessment

Each resolved error is assessed regarding its potential effect on the results. If the effect is anything more than negligible, its discovery and resolution are reported to the project participants via email. Similarly, if the error could have had an effect on the results of previous versions of the model, this is also reported.

## 4.7 Model Distribution

GoldSim models, like other computer model software, are open to modification. This is a benefit for modelers and researchers, since the logic is transparent and the model is easily maintained. This is a potential detriment to model integrity for the same reason. There are ways to tell if a model has been modified, however, as discussed above. Versioning and the tracking of all changes between versions are important. Nevertheless, developers and clients alike need to know the configuration status of the model they are using, and the read-only media-released versions always provide unambiguous starting points.

Release versions of the model(s) are ideally delivered to the client on read-only media (such as a CD-ROM), which inherently precludes modification of the models and supporting files. Using this method of delivery ensures that there is no ambiguity about the model and supporting documentation that constitutes the deliverable.

The GoldSim modeling software allows for complete construction and editing of models. The companion GoldSim Player, however, is currently available at no cost and can run GoldSim models that have been specifically "exported" as Player versions. The Player version of the model is not editable. For distribution to the general public, a GoldSim Player version of a Model can be provided as part of the deliverable. The Player model cannot be modified in its significant parts, though the user can still operate switches and controls to evaluate various effects.

# 5.0 References

GTG (GoldSim Technology Group), 2010a, *GoldSim User's Guide: Volumes 1 and 2*, GoldSim Technology Group LLC, Issaquah, WA.

GTG (GoldSim Technology Group), 2010b, *GoldSim Contaminant Transport Module User's Guide,* GoldSim Technology Group LLC, Issaquah, WA.

GTG (GoldSim Technology Group), 2010c, *Verification plan GoldSim version 10.50 SP1,* GoldSim Technology Group LLC, Issaquah, WA.

NRC (National Research Council), 2007, *Models in regulatory decision making*, National Research Council, The National Academies Press, Washington, D.C.

Sandia National Laboratories, 1995, *WIPP computer software requirements,* QAP19-1, Revision 0, Sandia National Laboratories Waste Isolation Pilot Plant, Carlsbad, NM.

# Appendix C: Neptune Check Print SOP

## Neptune and Company (Neptune) Internal Procedure

| | | |
|---|---|---|
| **General Procedure:**<br>Standard Operating Procedure (SOP)<br><br>**Contract Specific:**<br>Internal Neptune product | **Document No.**<br>NAC-0041_R2 | **Revision:**<br>2 |
| ***Document Status: Final*** | | |

| | |
|---|---|
| **Title:**<br>Neptune Check Print SOP | **Author:**<br>Michael Sully<br><br>**Revised by:**<br>Michael Sully |
| **Final Approval Signatures** | **Date** |
| **Corporate Quality Assurance Officer:**<br><br>**Printed Name:** Warren Houghteling<br><br>**Signature:** | |
| **Effective Date:** 1 Jan 2015 | |
| | |

**Document date Stamp:** 02 Feb 2015

# Neptune Check Print SOP
# for Verification of Data Entry

**Purpose**

This procedure describes the method for providing a check for the completeness and accuracy of data entry processes.

**Scope**

This procedure applies to manual or electronic data entry including data documentation packages developed for model input, databases or spreadsheets supporting models, and data/results tables included in reports.

**In this procedure**

This procedure addresses the following major topics:



## General information about this procedure

**Attachments**

This procedure has the following attachments:

| Number | Attachment Title | No. of pages |
|---|---|---|
| 1 | Check print 1 example | 1 |
| 2 | Check print 1 example data source document | 1 |
| 3 | Check print 2 example | 1 |

**History of revision**

This table lists the revision history and effective dates of this procedure

| Revision | Date | Description of Changes |
|---|---|---|
| 0 | 8 Sep 2004 | New document |
| 1 | 21 Dec 2010 | Revised signature page |

**Who requires training to this procedure?**

Personnel verifying data entry processes.

**Training method**

The training method for this procedure is on-the-job training by a previously trained individual and is documented by signature on training form and archived with project records

**Prerequisites**

None.

## Check print process

### Overview

This procedure applies to work processes requiring the manual entry or electronic transfer of data. Examples of entities that receive data include data documentation packages for model input parameters, external spreadsheets and databases used to provide input parameters for modeling, and tables of data/results in documents. Using this procedure, data entry or transfer is verified by comparing values in the receiving entity with values in the source documents/files to insure accuracy and completeness of the data entry or transfer. An individual other than the one compiling the data in the receiving entity should perform this check. For manual data entry 100 percent of the entries are checked. For electronic data transfer, 10 percent of the entries are checked. Inputs are checked using the check print process described below. This process can be used to verify most data entry tasks. Large files may require a modified procedure.

**Check print process**

To check print manually entered or electronically transferred data perform the following steps:

1. Obtain a paper copy of the receiving entity and a copy of the data source document. For example, see attachments 1 and 2.
2. Compare the parameter value in the source document including units with the value in the receiving entity to determine if it was entered accurately and completely.
3. If the value is correct, mark with a highlighter.
4. If the value is incorrect, circle in red ink and note the correct value.
5. Verify that the cited reference for the value is correct and complete with page number, table number, or other reference as required.
6. If the reference is accurate and complete, mark with a highlighter.
7. If the reference is inaccurate or incomplete, note corrections in red ink.
8. Label the checked receiving entity as “Check Print 1”, sign, date and return to the author for corrections.
9. When the corrections to the receiving entity are completed follow the same process as described in Steps 1 through 7, however, only the corrected values/references identified in check print 1 need to be checked. See attachment 3.
10. Label this check print as “Check Print 2”. Date and sign.
11. Repeat this process until all data/references entered are accurate and complete. The check print number is incremented for each iteration. Keep all iterations for archiving.

## Records resulting from this procedure

**Records**

The following records are created as a result of this procedure. Paper or electronic copies are maintained at Neptune and Company as described in the QAPP.

- All check prints
- Data source documents (or relevant sections thereof)

# Attachment 1

## An Example GoldSim Parameter List – Check Print 1

### 1.0 \DoseAssessment\PlantCRFood

Plant/soil concentration ratios are taken from Kennedy and Strenge (1992) [Table 6.16 p. 6-25]. All values in the table are defined as geometric means. The following table presents geometric mean values for four different plant parts and for each chemical element. These values are also used in plant-induced contaminant transport calculations (see the container \TransportProcesses\PlantTransport\PlantCRTransport).

| element | Leafy Veg (Ci/kg dry Plant) per (Ci/kg dry Soil) | Root (Ci/kg dry Plant) per (Ci/kg dry Soil) | Fruit (Ci/kg dry Plant) per (Ci/kg dry Soil) | Grain (Ci/kg dry Plant) per (Ci/kg dry Soil) |
|---|---|---|---|---|
| C | 7.00E-01 | 7.00E-01 | 7.00E-01 | 7.00E-01 |
| Cl | 7.11E+01<br>7.00E+01 | 7.00E+01 | 7.00E+01 | 7.00E+01 |
| Ar | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |


**Reference**

Kennedy, W.E. Jr., and D.L. Strenge, 1992. *Residual Radioactive Contamination From Decommissioning*, NUREG\CR-5512, Vol. 1 Pacific Northwest Laboratory, Richland, Washington.


Check Print 1 8 Sep 2004

Mary Jones

*An error was found for the entry for Cl for Leafy veg. The incorrect value was marked in red and the correct value was noted directly below it. The reference was determined to be incomplete since the data source was a single table in a 376 page document. The specific location of the table used as the data source was noted in red ink. The copy is labeled as Check Print 1 and is signed and dated by the reviewer.*

# Attachment 2

## Source Document Referenced in the Parameter List
## Kennedy and Strenge (1992)

**Table 6.16 Soil-to-plant concentration factors**

| | Soil-to-plant concentration factors (pCi/kg dry weight per pCi/kg soil) | | | |
|---|---|---|---|---|
| **Element/atomic number** | **Leafy vegetables** | **Root vegetables** | **Fruit** | **Grain** |
| H 1 | (-)* | (-)* | (-)* | (-)* |
| Be 4 | 1.0E-2 | 1.5E-3 | 1.5E-3 | 1.5E-3 |
| C 6 | 7.0E-1 | 7.0E-1 | 7.0E-1 | 7.0E-1 |
| N 7 | 3.0E+1 | 3.0E+1 | 3.0E+1 | 3.0E+1 |
| F 9 | 6.0e-2 | 6.0E-3 | 6.0E-3 | 6.0E-3 |
| Na 11 | 7.5E-2 | 5.5E-2 | 5.5E-2 | 5.5E-2 |
| Mg 12 | 1.0E+0 | 5.5E-1 | 5.5E-1 | 5.5E-1 |
| Si 14 | 3.5E-1 | 7.0E-2 | 7.0E-2 | 7.0E-2 |
| P 15 | 3.5E+0 | 3.5E+0 | 3.5E+0 | 3.5E+0 |
| S 16 | 1.5E+0 | 1.5E+0 | 1.5E+0 | 1.5E+0 |
| Cl 17 | 7.0E+1 | 7.0E+1 | 7.0E+1 | 7.0E+1 |
| Ar 18 | (-)** | (-)** | (-)** | (-)** |

* Concentration factors for $^{3}H$ are not needed because a special model is used to determine $^{3}H$ uptake in plants.

** Noble gas radionuclides are not assumed to be taken up by plants.

# Attachment 3

## An example GoldSim Parameter List—Check Print 2

### 2.0 \DoseAssessment\PlantCRFood

Plant/soil concentration ratios are taken from Kennedy and Strenge (1992) [Table 6.16, p. 6-25]. All values in the table are defined as geometric means. The following table presents geometric mean values for four different plant parts and for each chemical element. These values are also used in plant-induced contaminant transport calculations (see the container \TransportProcesses\PlantTransport\PlantCRTransport).

| element | Leafy Veg (Ci/kg dry Plant) per (Ci/kg dry Soil) | Root (Ci/kg dry Plant) per (Ci/kg dry Soil) | Fruit (Ci/kg dry Plant) per (Ci/kg dry Soil) | Grain (Ci/kg dry Plant) per (Ci/kg dry Soil) |
|---|---|---|---|---|
| **C** | 7.00E-01 | 7.00E-01 | 7.00E-01 | 7.00E-01 |
| **Cl** | 7.00E+01 | 7.00E+01 | 7.00E+01 | 7.00E+01 |
| **Ar** | 0.00E+00 | 0.00E+00 | 0.00E+00 | 0.00E+00 |

**Reference**

Kennedy, W.E. Jr., and D.L. Strenge, 1992. *Residual Radioactive Contamination From Decommissioning*, NUREG\CR-5512, Vol. 1 Pacific Northwest Laboratory, Richland, Washington.

Check Print 2 8 Sep 2004

________________________
Mary Jones

*The correction of the entry for Cl for Leafy veg and the additional data source information are verified and marked. The check print number is incremented and the copy is signed and dated by the reviewer. This is the final check print since the document is now accurate and complete.*