



---

Theses and Dissertations

---

2015-07-01

## A Cloud-Based GSSHA Index Map Editor Utility for Watershed Decision Support

Jocelynn Marie Anderson  
Brigham Young University - Provo

Follow this and additional works at: <https://scholarsarchive.byu.edu/etd>



Part of the [Civil and Environmental Engineering Commons](#)

---

### BYU ScholarsArchive Citation

Anderson, Jocelynn Marie, "A Cloud-Based GSSHA Index Map Editor Utility for Watershed Decision Support" (2015). *Theses and Dissertations*. 5292.  
<https://scholarsarchive.byu.edu/etd/5292>

This Thesis is brought to you for free and open access by BYU ScholarsArchive. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of BYU ScholarsArchive. For more information, please contact [scholarsarchive@byu.edu](mailto:scholarsarchive@byu.edu), [ellen\\_amatangelo@byu.edu](mailto:ellen_amatangelo@byu.edu).

A Cloud-Based GSSHA Index Map Editor Utility  
for Watershed Decision Support

Jocelynn Marie Anderson

A thesis submitted to the faculty of  
Brigham Young University  
in partial fulfillment of the requirements for the degree of  
Master of Science

Norman L. Jones, Chair  
E. James Nelson  
Daniel P. Ames

Department of Civil & Environmental Engineering  
Brigham Young University  
July 2015

Copyright © 2015 Jocelynn Marie Anderson

All Rights Reserved

## ABSTRACT

### A Cloud-Based GSSHA Index Map Editor Utility for Watershed Decision Support

Jocelynn Marie Anderson  
Department of Civil & Environmental Engineering, BYU  
Master of Science

Preventing damages from flooding is critically important for city managers and planners. Efforts in protecting infrastructure from flooding are often coupled with building hydrologic models to provide predictions of what is likely to happen during storm events. As land use changes, these models must be updated, which is more challenging with sophisticated models. A team of researchers from universities in Utah and Wyoming have been developing tools for water management in the Intermountain West as part of a collaborative NSF research grant called CI-WATER. In particular, a free and open source web platform called Tethys has been developed to support the development and hosting of hydrologic web applications. Tethys was used to develop a prototype application that uses a GSSHA runoff model and allows users to change land-use inputs to simulate the impact on a watershed for any type of land use change. The application also provides a method to run the edited model and produces a comparison report of before-and-after runoff and water depth as part of a decision-support framework.

Keywords: GSSHA, modeling, water, Tethys, GsshaPy, web application

## ACKNOWLEDGEMENTS

Firstly, I would like to thank Brigham Young University and all the wonderful professors I have had here. I have learned so much and have been inspired to continue learning eternally. I would also like to thank my family for encouraging me in my desire to pursue a master's degree and for supporting me in my study of civil engineering. I especially want to thank my husband Nathan for providing needed support while I worked on this research.

I would like to thank Dr. E. James Nelson for introducing me to the CI-WATER team and encouraging me to contribute to this project. I would also like to thank Dr. Norman L. Jones for all of his work as my committee chair and for meeting with me and spending so much time reviewing my thesis. Additionally, I would like to thank Dr. Daniel P. Ames ("Dan") for all he has taught me concerning hydroinformatics and programming and for helping me revise my thesis.

I would also like to thank Nathan Swain for all of his work in creating the Tethys Platform, which I used to build my app. Whenever Tethys didn't have functionality that I needed, he rapidly created an update with the needed functionality. He also provided hours of support and explanations while I prepared the GSSHA Index Map Editor App.

This research is based upon work supported by the National Science Foundation under Grant No. 1135483.

## TABLE OF CONTENTS

<b>LIST OF FIGURES .....</b>	<b>vii</b>
<b>1 Introduction.....</b>	<b>1</b>
1.1 Background.....	2
1.2 Tools for Hydrologic Modeling Applications.....	5
1.2.1 Tethys Platform.....	5
1.2.2 Gridded Surface Subsurface Hydrologic Analysis (GSSHA) Models.....	6
1.2.3 GsshaPy.....	7
1.2.4 PostGIS .....	7
1.3 Objective.....	8
<b>2 Methods.....</b>	<b>9</b>
2.1 Overview of Application Workflow .....	10
2.1.1 Opening the GSSHA Index Map Editor App .....	10
2.1.2 Selecting Models Currently Being Edited .....	12
2.1.3 Selecting an Index Map to Edit.....	13
2.1.4 Editing Index Maps.....	16
2.1.5 Editing Index Values.....	21
2.1.6 Submitting a Model.....	23
2.1.7 Running Models.....	24
2.1.8 Model Results .....	25
2.2 Software Organization .....	27
2.3 Work Flow .....	27
2.3.1 Initialization .....	28
2.3.2 Selecting a Model .....	28

2.3.3	Loading Model to GsshaPy Database .....	29
2.3.4	Editing the Raster Values of Index Maps .....	30
2.3.5	Updating Index Variable Values.....	31
2.3.6	Submitting the Altered Model .....	32
2.3.7	Viewing Edited and Submitted Models .....	32
2.4	Deleting Models from the Database .....	33
2.5	Running Models.....	33
2.6	Viewing Model Results .....	34
<b>3</b>	<b>Results .....</b>	<b>35</b>
3.1	Introduction to the Test Case .....	35
3.2	GSSHA Model Assumptions.....	35
3.3	Test Case Analysis.....	36
3.4	Test Case 1 .....	36
3.4.1	Results of Test Case 1 .....	38
3.5	Test Case 2.....	40
3.5.1	Results of Test Case 2.....	41
3.6	Overview of Results.....	43
<b>4</b>	<b>Conclusions.....</b>	<b>44</b>
4.1	Review of Objectives.....	44
4.1.1	Simple Mechanism for Editing Index Maps .....	44
4.1.2	Integrate Index Map Editor into Web-Based Modeling System.....	45
4.1.3	Demonstrate How the System Can Be Implemented.....	45
4.1.4	Explore the Capability of Tethys in Meeting Needs.....	46
4.1.5	Test the Developed System in a Case Study.....	47
4.1.6	Recommendations Regarding Use of Tethys for Similar Modeling Applications ...	47

4.2	Opportunities for Continued Development.....	48
4.3	Software Availability.....	49
<b>REFERENCES.....</b>		<b>50</b>

## LIST OF FIGURES

Figure 1-1: Tethys Platform Diagram.....	6
Figure 2-1: Overview of the app workflow for the user. ....	10
Figure 2-2: The GSSHA Index Map Editor app in the Tethys App Library. ....	11
Figure 2-3: Home page with drop down menu for selecting a GSSHA model. ....	12
Figure 2-4: Home page when no GSSHA models are in the CKAN datasets. ....	13
Figure 2-5: Example of a “combo” index map. ....	14
Figure 2-6: Example of a “luse” index map.....	14
Figure 2-7: Example of a “soil” index map. ....	15
Figure 2-8: Methods of editing index maps.....	15
Figure 2-9: Editing the index map by drawing a polygon and changing the ID.....	17
Figure 2-10: Two completed polygons that have been drawn.....	17
Figure 2-11: Index map after changes have been saved. ....	18
Figure 2-12: Sample of an uploaded shapefile. ....	19
Figure 2-13: Page for combining index maps or replacing with an existing index map. ....	20
Figure 2-14: Results of combining index maps versus the original index map.....	21
Figure 2-15: Example of variable values being edited for an index map. ....	22
Figure 2-16: Index mapping table values are reviewed before returning to the index map selection page.....	23
Figure 2-17: Modal with information for submitting a model.....	24
Figure 2-18: Menu of submitted projects. The first one has not been run, while the second has been and results have been returned. ....	25
Figure 2-19: Comparison hydrograph for before and after the edits were made.....	26
Figure 2-20: Menu with depth map options and depth map display.....	26
Figure 2-21: Tethys file scaffold applied to new projects and intended contents of files. ....	27



Figure 2-22: Sample code used to change the index map values based on the geometry provided. ....	30
Figure 2-23: Sample code used to change index map values based on the GeoJSON provided. ....	31
Figure 3-1: Test case 1 – Development on eastern side of the watershed. ....	37
Figure 3-2: Input roughness values for Test Case 1. ....	37
Figure 3-3: Comparison hydrograph for Test Case 1. ....	39
Figure 3-4: Resulting maximum depth maps for Test Case 1 showing less maximum depth on the eastern side of the watershed after the roughness was lowered. ....	39
Figure 3-5: Test Case 2 shapefiles showing development. ....	40
Figure 3-6: Test Case 2 – Land use index map before and after updating by shapefile. ....	41
Figure 3-7: Comparison hydrograph for Test Case 2. ....	42
Figure 3-8: Resulting maximum depth maps for Test Case 2 showing a lower maximum depth in the areas of development. ....	42

## 1 INTRODUCTION

Preventing damages from flooding is critically important for city managers and planners. Efforts in protecting infrastructure from flooding are often coupled with building hydrologic models to provide predictions of what is likely to happen as a result of storms and extreme events. Many models predict the runoff or flooding that will occur due to hypothetical storm events under certain land conditions. As cities develop and land uses change, these models are then able to predict changes in runoff. This can aid in preventing major damages from events such as floods and are used to guide municipal planning in terms of runoff and flood control infrastructure.

Predicting changes in runoff due to urbanization is a challenging, but critical task. We gain confidence in a model's prediction capability based on how accurately the model can reproduce past events, but this often requires sophisticated models that have undergone many hours of calibration. An additional challenge is the constant changes in land use requiring frequent updates to prepared models. Historically, this is done through taking a calibrated model, making changes to the input files, and rerunning it. This often involves outsourcing the work to an external consulting firm or maintaining a staff of highly-trained modelers, which is often costly and time consuming. In some cases this may mean that cities must make decisions that are not based on high-resolution models.

One way to address these challenges is to develop simple cloud-based tools for decision support that involve a simple web front end and a sophisticated model on the backend where the

model runs as part of a scripted process. This allows city managers and planners to use sophisticated models without needing to be modeling experts themselves. They will face the initial cost of setting up the model or system, but then will experience cost savings and better decision-making.

These cloud-based tools would also be beneficial for educational purposes. Students can use them to learn what effect land use change has on runoff by making edits to models and comparing the results. For students without access to expensive software licenses, this would be beneficial in learning about and visualizing the effects of changing land use.

Another projected user for a cloud-based index map editor application is an organization such as the forest service. When fires produce hydrophobic ground, a model can be updated to account for the changes in these areas. Flooding concerns can then be predicted and addressed using the new characteristics of the burned area to update the model.

## 1.1 Background

Various Decision Support Systems (DSS) have been developed for watershed management. Spatial Decision Support Systems (SDSS) are DSS's that integrate spatial elements. These are especially useful in watershed management tools since watershed information varies spatially, and gridded GIS data in combination with non-spatial data can then be used in computations (Choi 2005). Not only can GIS data be stored and manipulated for computations, but an SDSS is also capable of displaying results in a mapped format so they can be more easily interpreted by decision-makers (Wilson 2000).

The Internet and cloud technology have provided a powerful mechanism to store and share spatial data. Through the work of the Open Geospatial Consortium (OGC), there is a standardized method of storing and sharing geographic information on web pages. Geographic

Markup Language (GML), which is based off of EXtensible Markup Language (XML), is the format in which spatial objects and properties are stored (Choi 2005). This uniformity in format allows data to be shared and understood by different people and programs.

Web-based SDSS allow users to access tools and information through a web browser. The user has an interface that includes maps, menus, and forms, while all of the computation takes place on the web server. After the server finishes computations, it is able to return the results or files to the user through the web interface (Choi 2005).

One benefit of web-based SDSS compared to a desktop application or a command line model is that no software needs to be downloaded to the users computer. It is capable of being run on any computer or mobile device with internet access and no information has to be stored on their device. Processing also takes place on a server, so the computing power of the user's device does not have an effect on how long a model takes to run. Additionally, the application can protect the original model from harmful edits by not allowing the user to override the original model, but forcing all changes to be made on a copy.

There are a number of web-based watershed modeling tools with different designs and purposes that have been developed and are hosted on the Internet. Some examples of web-based watershed modeling tools are WHYGIS and L-THIA.

Watersheds can be delineated using the location of an outlet in conjunction with a digital elevation model (DEM). The Web-based HYdrologic Geographic Information System (WHYGIS) is a system that performs online watershed delineation. The system uses GIS capabilities to perform real-time watershed delineation based off an outlet location defined by the user and submitted through the web-GIS user interface. This delineation can then be used to clip other gridded data, such as soil and land use rasters. These rasters can then be used to

calculate a Natural Resources Conservation Service (NRCS) curve number map that can be used in calculating runoff (Choi 2005).

One example of integration of both building and executing models via a web interface is the Long-Term Hydrologic Impact Assessment (L-THIA) tool developed by an interdisciplinary team at Purdue University. L-THIA is a spreadsheet-based program, so the initial web interface allows users to select land use, soil type, and the area of each combination. The inputs are used in calculations to produce results of runoff and pollution (Pandey 2000).

The team also incorporated GIS into the L-THIA model. Using the WHYGIS application, users are able to specify an outlet location and a watershed is generated using elevation data. Soil and land use data are also extracted and used to generate NRCS curve numbers, which are then displayed for the user. The user can then run the model without having any data beforehand other than the location of the outlet of the watershed (Choi 2005). These models provide insight into long-term hydrologic conditions, such as pollution and infiltration rather than an extreme storm (Pandey 2000).

Both WHYGIS and L-THIA are useful hydrologic modeling tools, but they lack important features. WHYGIS adds additional functionality to L-THIA so that watershed areas and models can be developed on the web, but L-THIA does not allow users to spatially edit land use areas. The model is also not spatially distributed and is based off basic spreadsheet calculations. There is a need for capability of running and editing a more complex model, which accounts for spatial distribution, through a web interface.

## 1.2 Tools for Hydrologic Modeling Applications

There are many tools available for developing web-based hydrologic modeling applications. A few tools that were used in developing the GSSHA Index Map Editor Application will be discussed in the following sections.

### 1.2.1 Tethys Platform

The CI-WATER group, comprised of researchers from universities in Utah and Wyoming, have been developing high-performance computer modeling and computation resources to simulate factors facing water management in the Intermountain West (CI-WATER, 2015). One of the projects being developed by the CI-WATER team is a free and open source web platform called Tethys Platform for developing and hosting water resources web applications (Swain, Tethys, 2014).

Tethys Platform is a free and open source web platform developed at Brigham Young University for developing and hosting hydrologic web applications. Tethys is built on the Django Python web framework. It includes a software development kit that allows users to develop web apps using the Python programming language. Users can develop their own hydrologic web applications and create a library of apps that can be accessed anywhere with an internet connection. Tethys apps are built using the Model View Controller (MVC) architectural approach. Tethys simplifies user interface development with a number of elements called gizmos that allow users to easily create user interface elements such as interactive maps, message boxes, buttons, and inputs.

Tethys Platform includes a software suite with a number of free and open source software packages that provide the necessary GIS functionality of water resource web applications. The software suite includes PostgreSQL with the PostGIS extension that allows for storage of spatial

data in a database. It also includes Geoserver which can be used to publish spatial data and 52 North Web Processing Service which can be used to perform geoprocessing tasks (Swain, Tethys, 2014). A summary of the major components of Tethys Platform is shown in Figure 1-1.

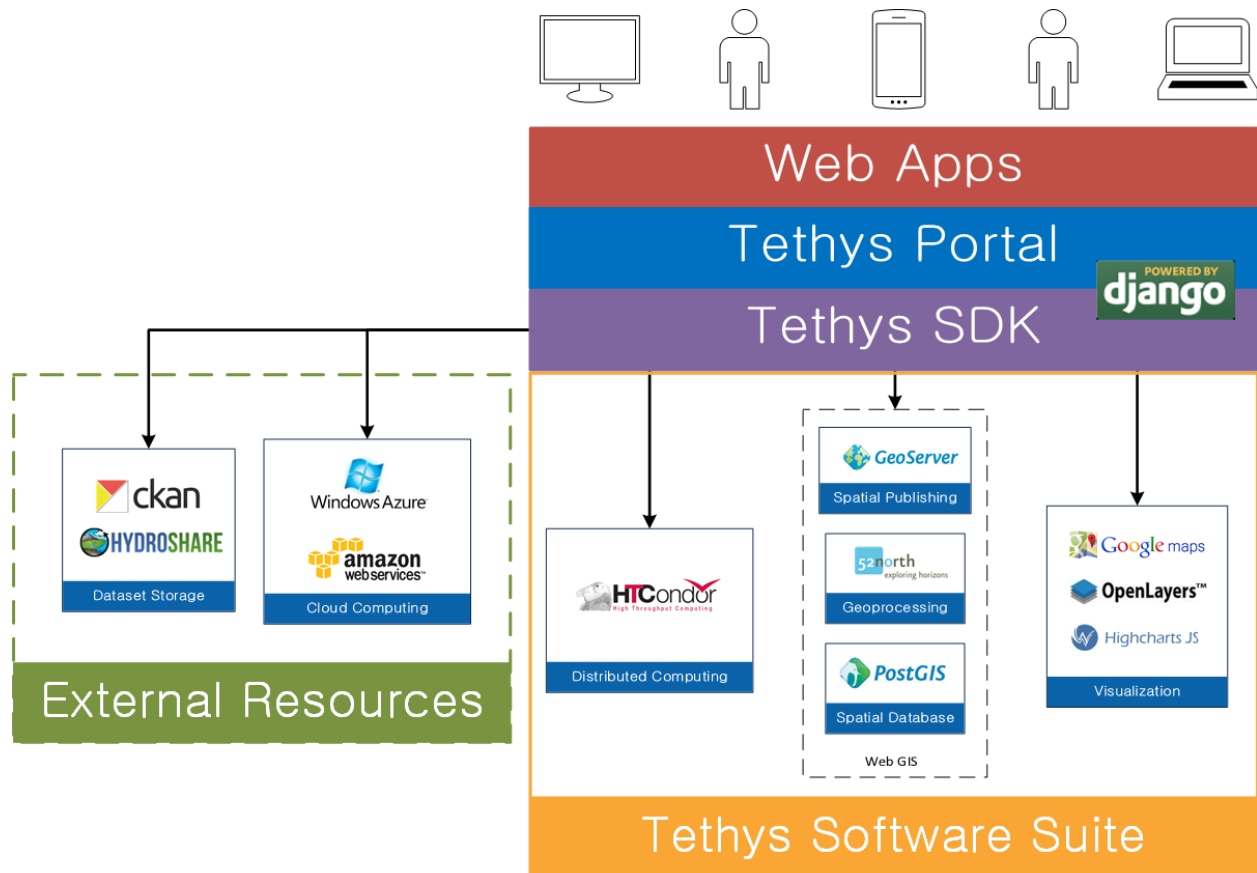


Figure 1-1: Tethys Platform Diagram

### 1.2.2 Gridded Surface Subsurface Hydrologic Analysis (GSSHA) Models

Gridded Surface Subsurface Hydrologic Analysis (GSSHA) models are physically-based watershed analysis models that simulate runoff processes such as stream routing, groundwater flow, exfiltration, precipitation distribution, precipitation interception, evapotranspiration, and infiltration (Downer, 2004).

The US Army Corp of Engineers recommends using the Watershed Modeling System (WMS) interface for developing input files for GSSHA as well as for viewing the output files.

WMS is able to use Geographic Information System (GIS) data to produce GSSHA specific raster files. GSSHA stores this spatially distributed data using the GRASS ASCII data file format. These files are used to assign values to index maps in the models (Downer, 2003).

GSSHA models are especially applicable to studies involving land-use change because of the physical basis of the model and the way in which parameters can be distributed (Downer, 2004). Index maps contain data for properties such as land use and soil type, which are then used in analyzing runoff, stream routing, overland and stream sediment processes, and constituent transport (Downer, 2006).

### **1.2.3 GsshaPy**

GsshaPy is an object relational model (ORM) for GSSHA models. It is built on top of the SQLAlchemy ORM and exposes GSSHA files to a web development environment by reading them into an SQL database (Swain, GsshaPy, 2014). The files can then be easily altered programmatically, after which the information in the database can be written back out to file for execution. GsshaPy also uses MapKit, a Python module, to provide methods for generating KML visualizations of the raster and vector model files.

### **1.2.4 PostGIS**

PostGIS is a spatial database extender for PostgreSQL databases (PostGIS, 2015). It allows querying and manipulation of spatial data that is stored in PostgreSQL databases. Included in PostGIS is a library of functions that edit, analyze, and process raster objects that are in a PostgreSQL database. Some of these functions include clipping by a given boundary, performing algebra between two rasters, setting the value of raster cells within a given boundary, and many more that allow for powerful data manipulation.



### 1.3 Objective

The research goals include:

1. Provide a relatively simple mechanism for editing index maps.
2. Integrate such an index map editor into a web-based modeling system to provide cross platform functionality.
3. Demonstrate how this system can be implemented in a web-based GIS decision support system.
4. Explore the capabilities of Tethys for meeting these needs.
5. Test the developed system in terms of a case study.
6. Make recommendations regarding the application of web-based GIS tools, specifically Tethys, for future development of similar modeling applications.

In the following sections I will describe the application workflow, the software organization, and the back-end workflow. I will then describe two test cases in which the GSSHA Index Map Editor app would be used and provide results from the tests. To conclude I will summarize the application, its uses and benefits, areas of continued development, and give a review of the Tethys Platform in developing hydrologic web applications.

## 2 METHODS

The GSSHA Index Map Editor App is designed to allow users to make edits to an existing GSSHA model. The GSSHA input files that are modified during a land use change analysis are index maps and the mapping table. Index maps are GRASS ASCII maps that represent spatially distributed properties of the model, such as land use or soil type. The mapping table file maps hydrologic properties to the indices of the index maps. Edits can be made to both the index map rasters themselves and the hydrologic properties used in the GSSHA model mapping tables.

To simplify the editing process, GsshaPy is used to read the model files into an SQL database. The changes are saved in the database and when edits are completed, the modified model is written back to file. All of the GSSHA input files are then placed into a zip archive for easy transport and archival. The GSSHA zip archive is then sent to be run on a web processing service. After the model is processed, the results are returned and displayed on a webpage for the user to compare the results from the original and edited models.

In the following sections I will give a more in-depth description of the design of the GSSHA Index Map Editor application. Firstly, I will give an overview of the application workflow and I will then discuss the software organization.

## 2.1 Overview of Application Workflow

The app was designed to provide a simple and straightforward workflow for the user. The process involves selecting a model, making edits to index maps, and submitting, running, and getting results for the updated model. An overview of the workflow is shown in Figure 2-1 and the workflow will be described in more detail in the following sections.

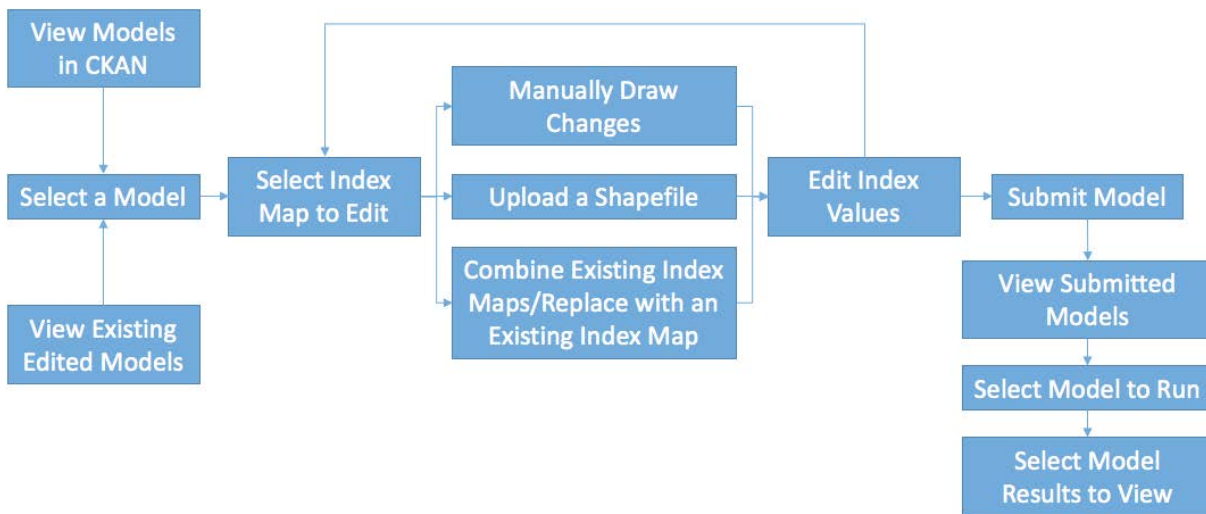
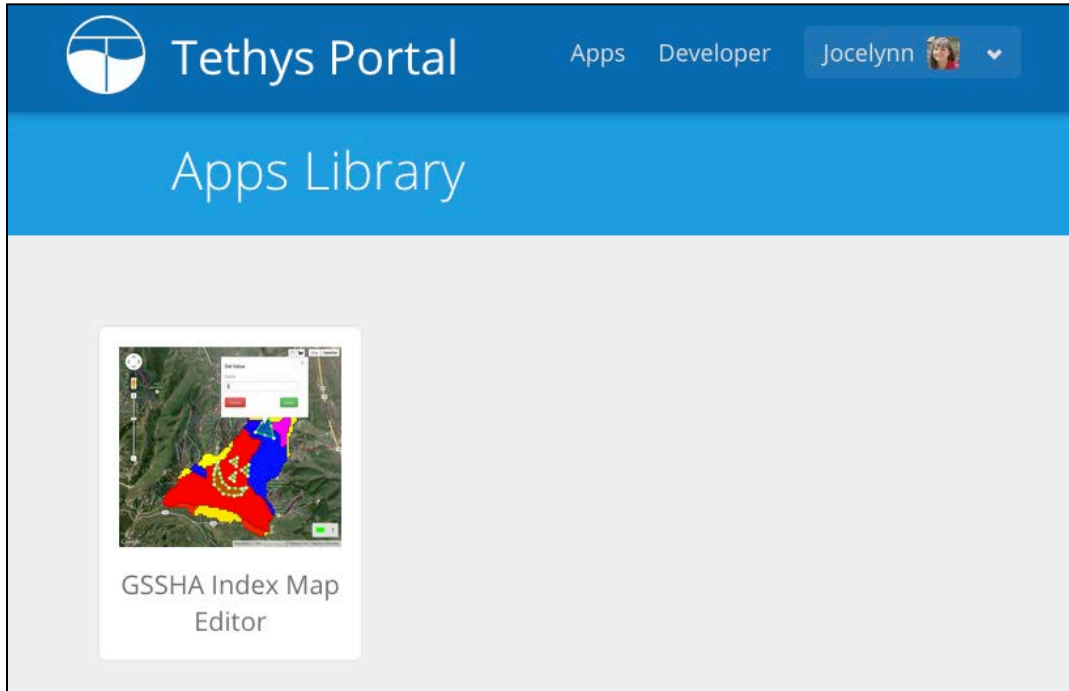


Figure 2-1: Overview of the app workflow for the user.

### 2.1.1 Opening the GSSHA Index Map Editor App

When the user opens Tethys, they will see an Apps Library. This is the location of all the apps being hosted on the server they are on. The server I used only had the GSSHA Index Map Editor app, as seen in Figure 2-2. When the square containing the app picture and title is clicked on, the app home screen is opened.



**Figure 2-2: The GSSHA Index Map Editor app in the Tethys App Library.**

From the home screen the user is able to select a model from the drop down menu to the left of the map as seen in Figure 2-3. The list is comprised of GSSHA models that are located in the CKAN database linked to the app. When a model is selected, a KML of the mask map is generated and displayed on the map. This shows the total area of the watershed that is analyzed in the model, giving the user a reference of the area they are working with. When there are no GSSHA models present, a message is displayed instructing the user to upload a model to the dataset as seen in Figure 2-4.

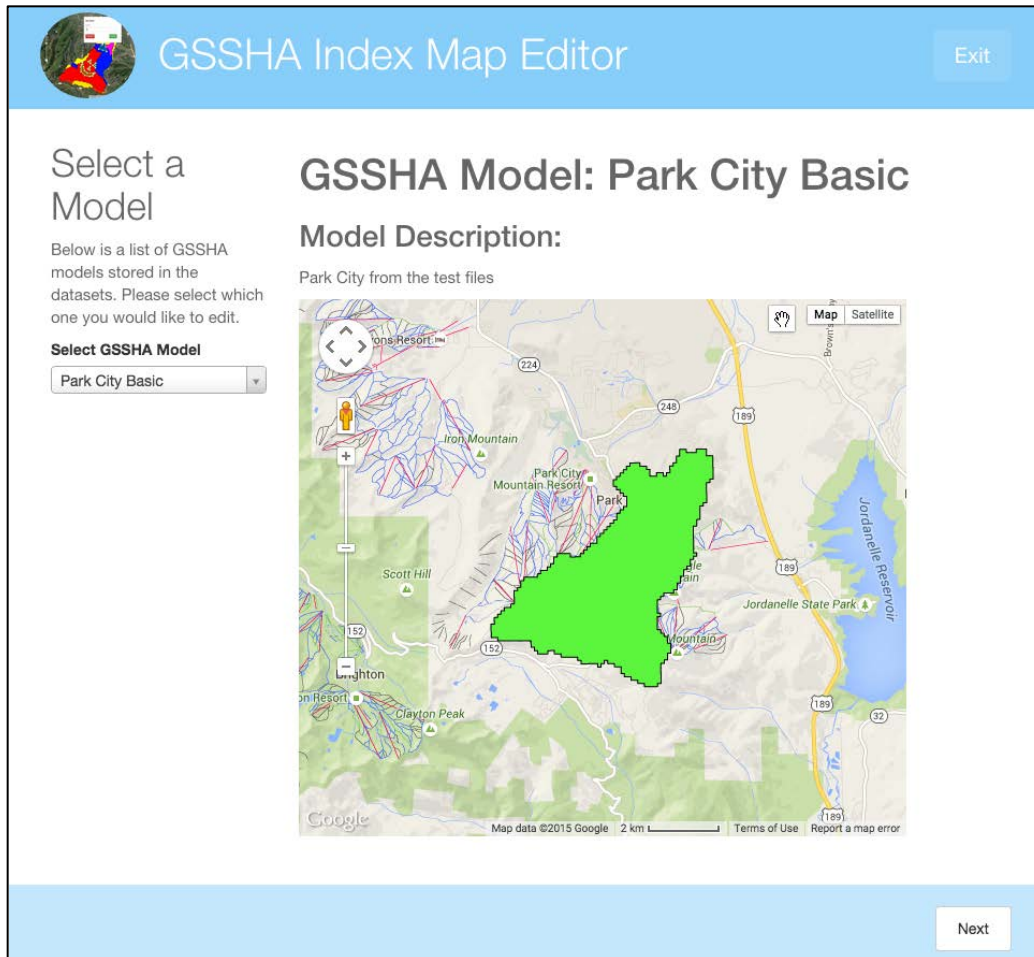
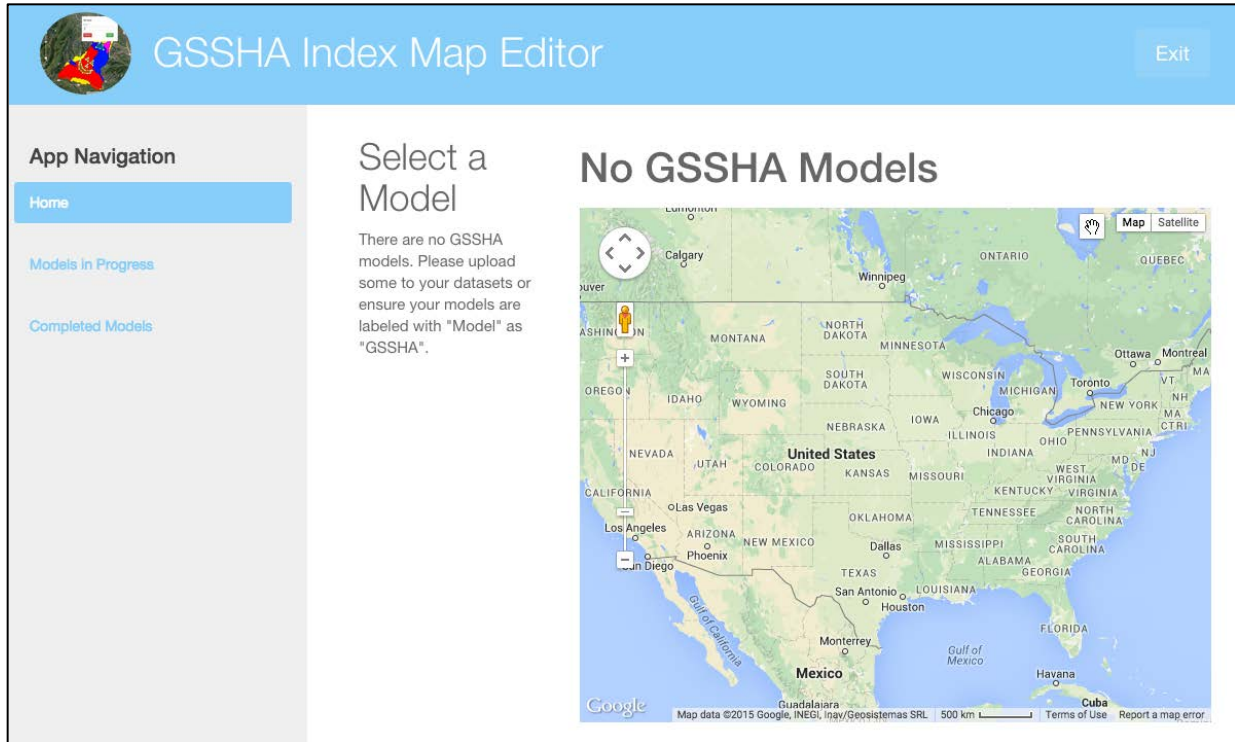


Figure 2-3: Home page with drop down menu for selecting a GSSHA model.

### 2.1.2 Selecting Models Currently Being Edited

The expandable menu located on the far left side of the app's web pages (Figure 2-4) allows for navigation to models that users have already started editing or to models that have already been submitted. The list only includes models that the specific user has created and allows the user to delete models that they no longer want to access.



**Figure 2-4: Home page when no GSSHA models are in the CKAN datasets.**

### 2.1.3 Selecting an Index Map to Edit

After the user selects a model and clicks “Next”, the index maps for the GSSHA model are displayed and the user can view each of them by selecting on the name of the index map to the left of the map as shown in Figure 2-5 through Figure 2-7. After highlighting the index map that needs to be edited, the user can click the “Edit Selected Index Map” button at the bottom of the page. This brings up a modal containing the options for editing the map (Figure 2-8).

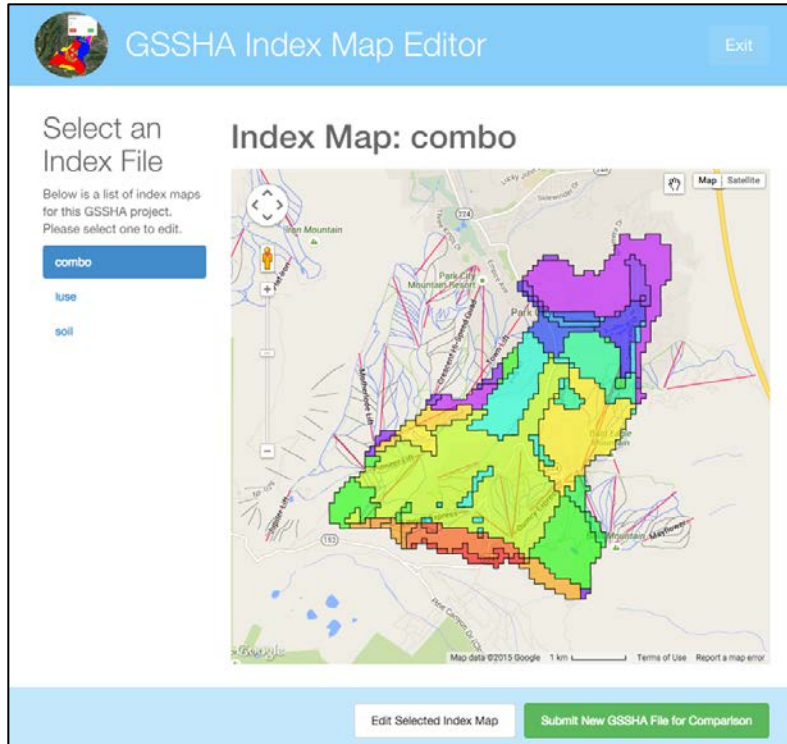


Figure 2-5: Example of a “combo” index map.

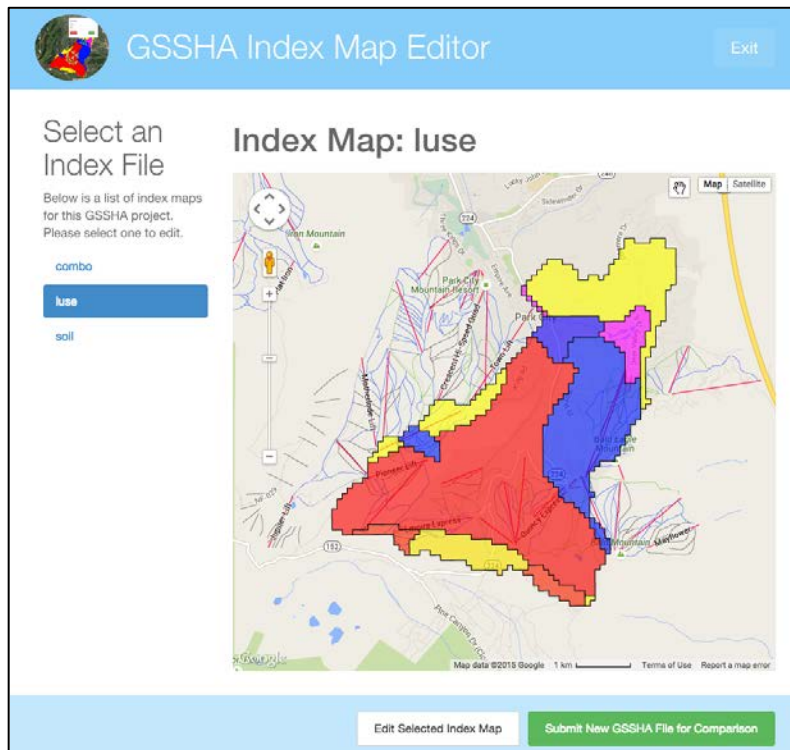


Figure 2-6: Example of a “luse” index map.



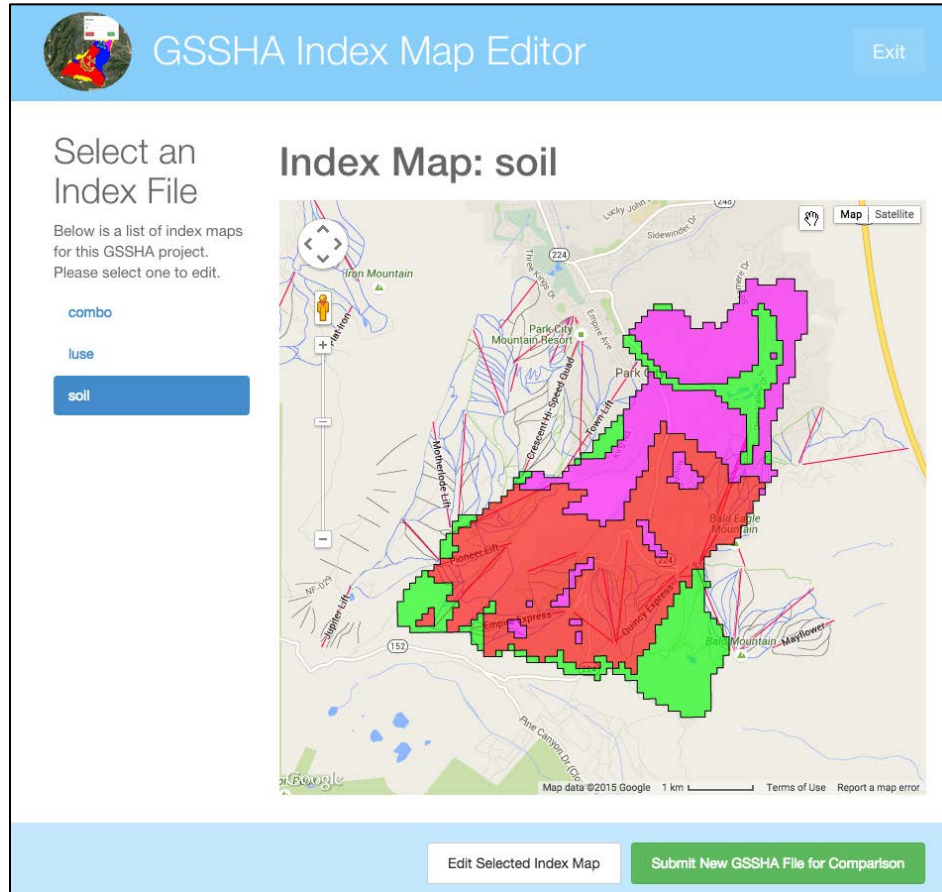


Figure 2-7: Example of a “soil” index map.

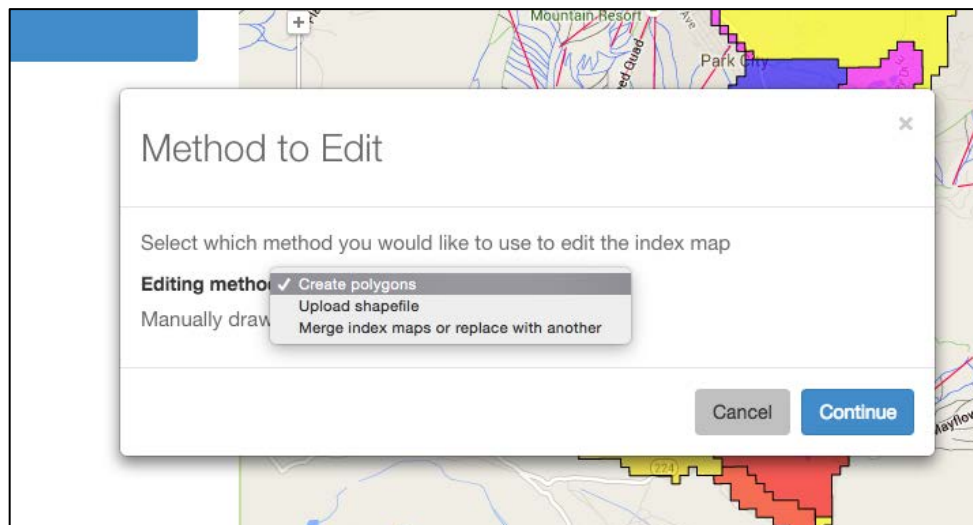


Figure 2-8: Methods of editing index maps.



## 2.1.4 Editing Index Maps

From the modal in Figure 2-8, the user can select from creating polygons, uploading a shapefile, or merging index maps/replacing with an existing index map. After selecting an option they are directed to the corresponding web page. The three methods for editing index maps will be discussed in the following sections.

### 2.1.4.1 Manually Drawing in Edits

When the option is selected for manually drawing in edits, the user is taken to an editable Google Map implemented via Tethys gizmos. This allows the user to draw a polygon, specify the ID, and save the new map. Multiple polygons with different IDs can be drawn before saving the index map. Figure 2-9 through Figure 2-11 demonstrate the process of editing a map with polygons. Figure 2-9 shows a polygon that is drawn in with an ID input box in the top left corner where the ID has been set to 12. Figure 2-10 shows another polygon drawn with an ID of 15. Figure 2-11 shows what the index map looks like after the changes are saved. The new index values are also added to the table and the user is able to edit descriptions for each of the IDs.

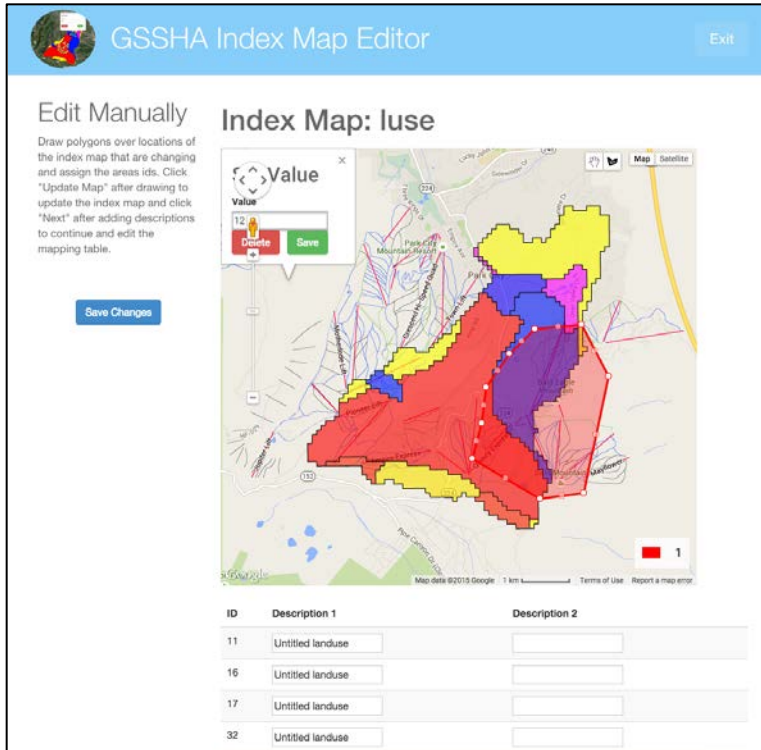


Figure 2-9: Editing the index map by drawing a polygon and changing the ID.

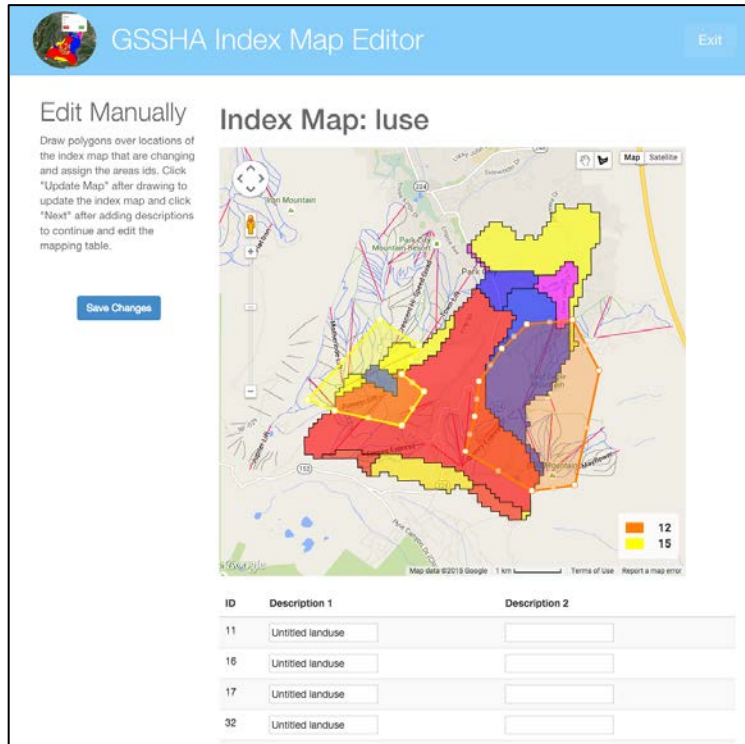
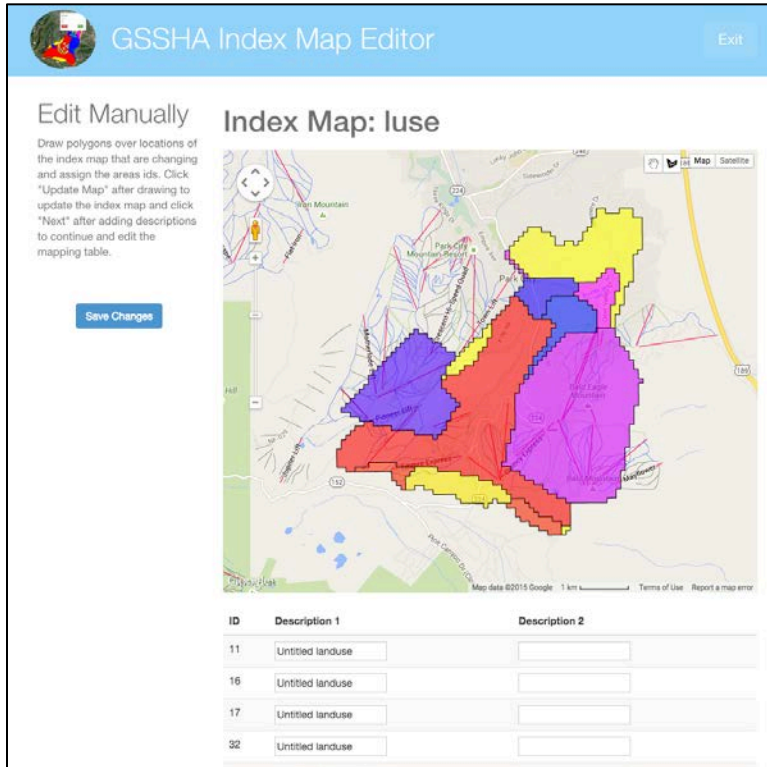


Figure 2-10: Two completed polygons that have been drawn.

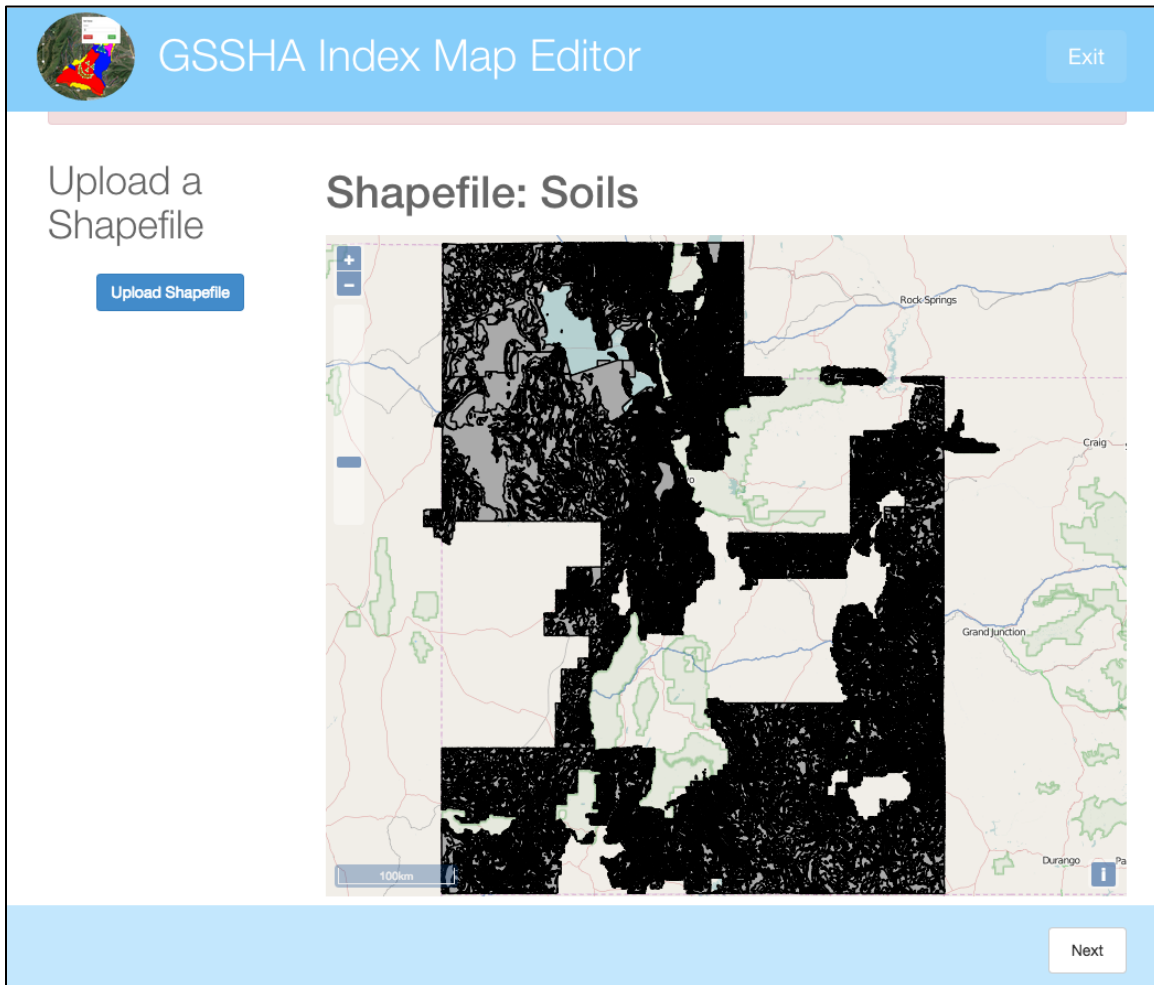


**Figure 2-11: Index map after changes have been saved.**

#### 2.1.4.2 Editing by Shapefile

When the user selects the option of editing by shapefile, they are taken to a page with a map and a button to upload a shapefile. When that is clicked, a modal appears and they are able to browse their computer files to find files relating to the shapefile to upload. It is required that they select a set of “.shx”, “.shp”, “.dbf”, and “.prj” files. This is important as the “.shp” file contains the binary geometry needed to define the boundaries, the “.dbf” file contains the feature attributes, the “.shx” file is an index file linking the “.shp” and “.dbf” files, and the “.prj” file defines the projection of the shapefile. It was decided that these files not be uploaded as a zipped file so that a check could be performed to ensure all the necessary files are present for the shapefile. When they upload all the required files, the files are zipped together and the shapefile is uploaded to GeoServer and then retrieved and displayed on the map (as shown in Figure 2-12).

When the next button is clicked, the values from the shapefile are used to replace the values in the index map and the user is able to edit the ID descriptions.



**Figure 2-12: Sample of an uploaded shapefile.**

### **2.1.4.3 Combine Two Index Maps or Replace with an Existing Index Map**

The last option for editing index maps is to combine two index maps, or to replace it with an existing index map. On this page (as shown in Figure 2-13) the user can view the index maps by clicking on the list on the left to change the map shown. Below the map is a menu where the user can select which index maps to use in replacing the index map being edited. In Figure 2-13 it is shown that the “combo” index map is being replaced by combining the “luse” and “soil” index maps. After the “Next” button is selected, the new index map is created as a combination

of the two selected index maps as shown in Figure 2-14. If the option “None” had been selected in the second box, the “combo” index map would be replaced by the “luse” index map.

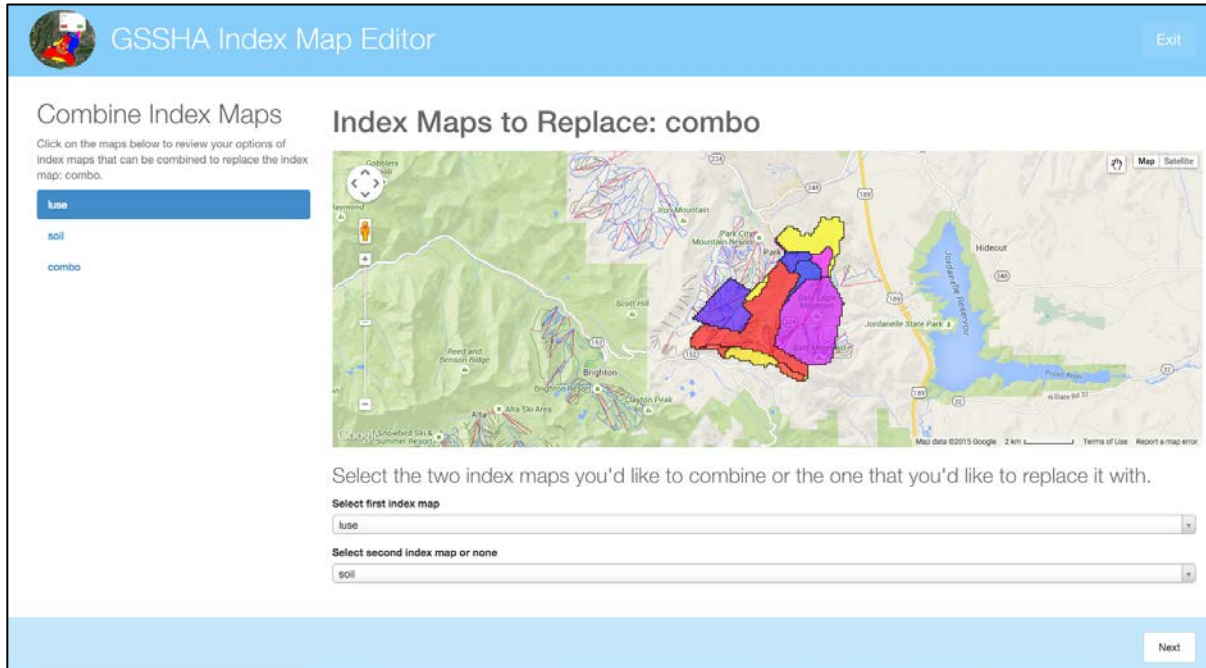


Figure 2-13: Page for combining index maps or replacing with an existing index map.

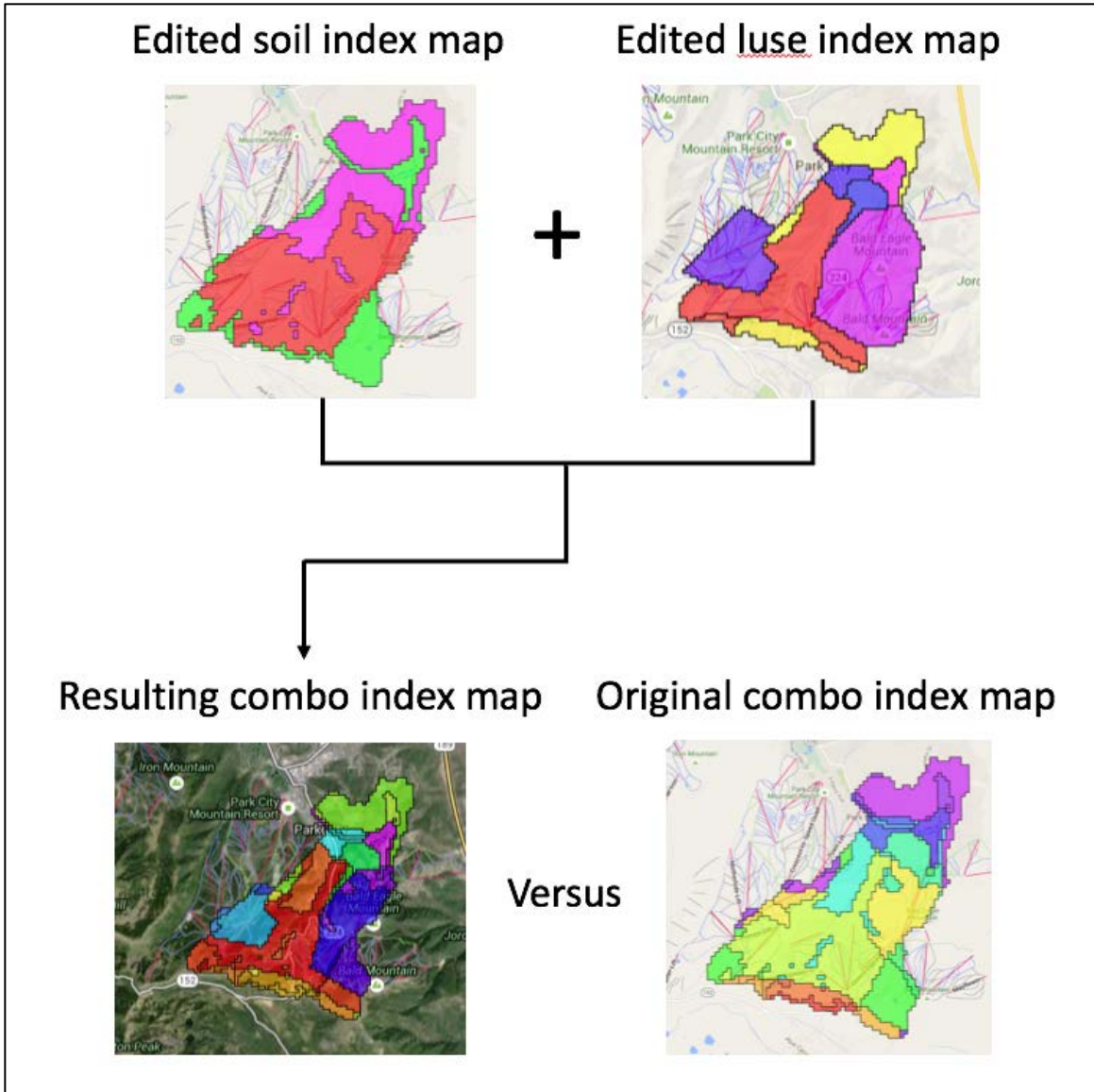


Figure 2-14: Results of combining index maps versus the original index map.

### 2.1.5 Editing Index Values

After an index map is edited in any of the before mentioned ways, the next page allows the user to change the variable values associated with each index map ID (as shown in Figure 2-15). These are referred to as mapping table values and after the user finishes making edits they are able to review their edits (pictured in Figure 2-16) before returning to the index map selection menu where they may select another index map to make edits to before submitting the model.



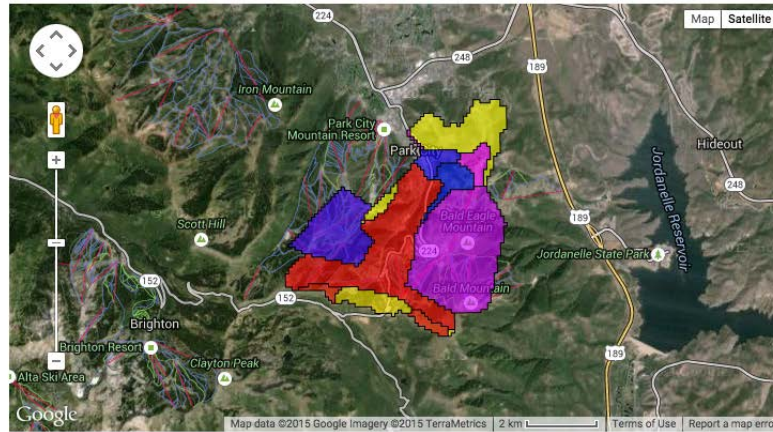


## Edit Index Mapping Table Values

Below is a list of mapping tables you can edit that are related to this index map. After making changes to the values, click "Next" to continue and review the mapping tables.

**ROUGHNESS**

## Index Map: luse



ID	Description 1	Description 2	ROUGH
11	Untitled landuse		<input type="text" value="0.011"/>
12	Added LU		<input type="text" value="0.001"/>
15	Second Added LU		<input type="text" value="0.002"/>
16	Untitled landuse		<input type="text" value="0.011"/>
17	Untitled landuse		<input type="text" value="0.011"/>
32	Untitled landuse		<input type="text" value="0.05"/>
33	Untitled landuse		<input type="text" value="0.04"/>
41	Untitled landuse		<input type="text" value="0.1"/>
42	Untitled landuse		<input type="text" value="0.15"/>

Figure 2-15: Example of variable values being edited for an index map.

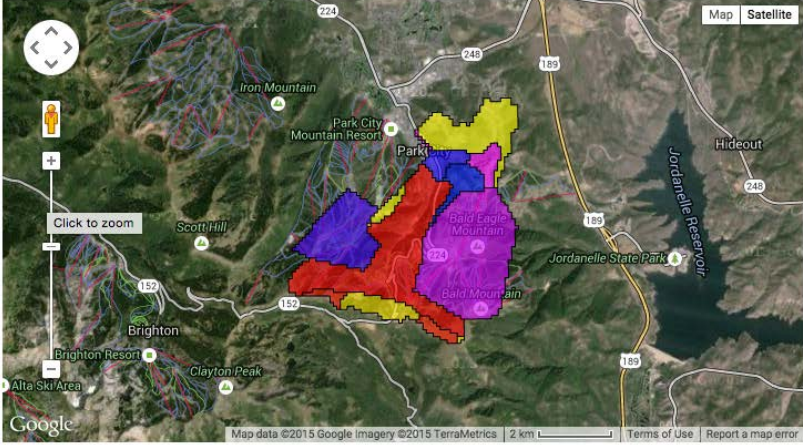
GSSHA Index Map Editor
Exit

### Review Index Mapping Table Values

Please review your edits to this index map's mapping tables and click "Next" to continue or "Back" to return to the editable mapping tables.

ROUGHNESS

## Index Map: luse



ID	Description 1	Description 2	ROUGH
11	Untitled landuse		0.011
12	Added LU		0.001
15	Second Added LU		0.002
16	Untitled landuse		0.011
17	Untitled landuse		0.011
32	Untitled landuse		0.05
33	Untitled landuse		0.04
41	Untitled landuse		0.1
42	Untitled landuse		0.15

Back: Edit Mapping Tables

Next: Save Index Map

**Figure 2-16: Index mapping table values are reviewed before returning to the index map selection page.**

### 2.1.6 Submitting a Model

When the user has completed making edits to the index maps, they can click “Submit New GSSHA File for Comparison” and a modal will appear as shown in Figure 2-17. The user may specify the name for the new model (which is used in saving the newly run model to



CKAN, allowing it to be used in future comparisons) and may optionally provide a description for the model.

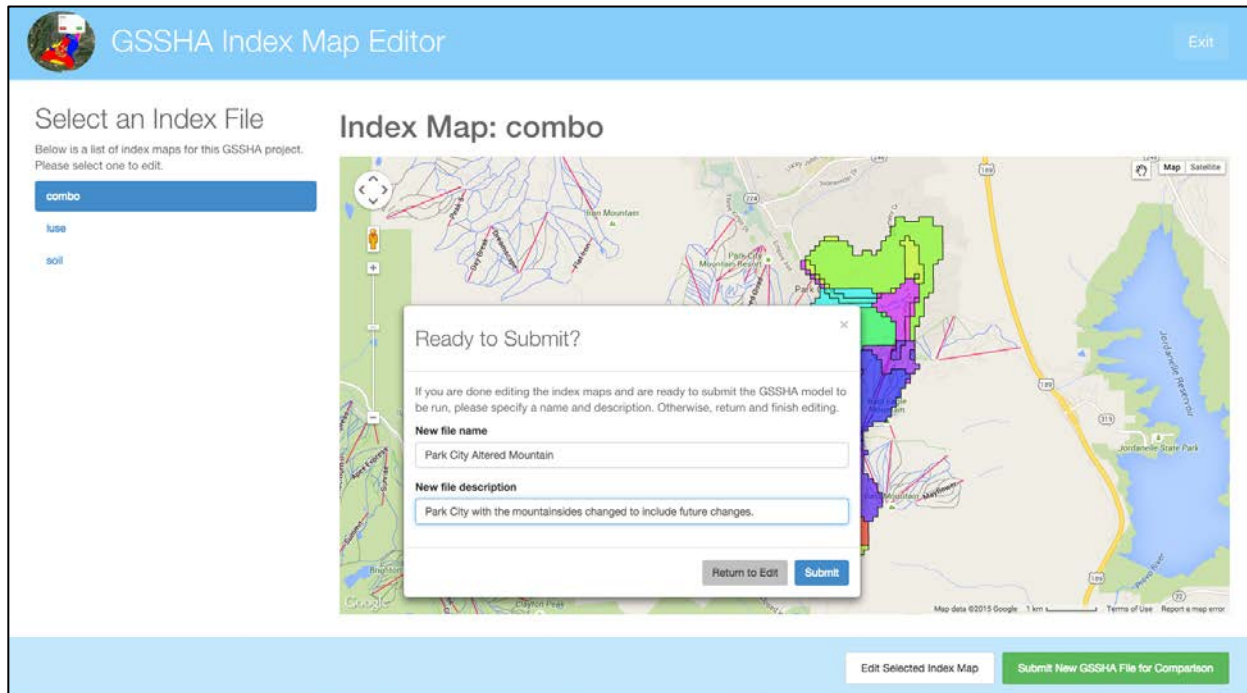
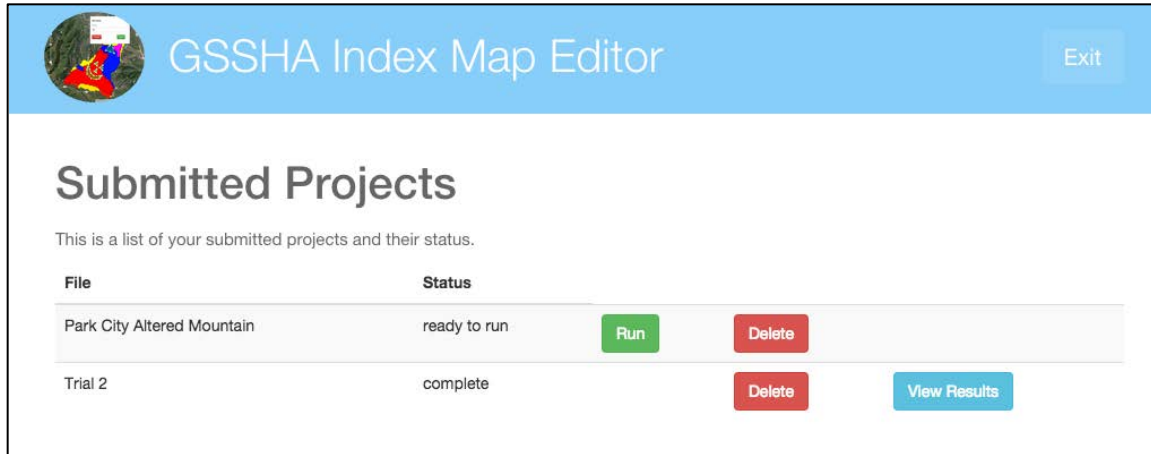


Figure 2-17: Modal with information for submitting a model.

### 2.1.7 Running Models

After submitting a model, the user is directed to a page containing all of the models they have submitted (as shown in Figure 2-18). Models that have not been run have the option of being run or deleted, while models that have been run can be deleted or the user can be redirected to a page to view the results. When the user clicks the run button, the models are submitted to the web processing service and the run button disappears as the status changes to “processing”. The submitted models include the one that has been edited by the user, and the original model that the edits were made on. After the results of the model runs are returned, the “View Results” button appears.



**Figure 2-18: Menu of submitted projects. The first one has not been run, while the second has been and results have been returned.**

### 2.1.8 Model Results

The results page has a number of elements. The first is a hydrograph. This is made using HighCharts through a Tethys gizmo and allows the user to turn the individual hydrographs from the original and newly edited models on or off. When the page loads both are visible, showing a comparison hydrograph as seen in Figure 2-19.

Below the hydrograph is a depth map display with a drop down menu for the user to select different depth map options (as shown in Figure 2-20). The options include the maximum depth map and time series depth map for both the original and altered models. The time series map is not viewable on the webpage, but there is a link to download the file and it can then be viewed using interfaces such as Google Earth. Below the map there is also a link to download the zipped result files for each model. These files are pulled from CKAN where they are stored for future use.

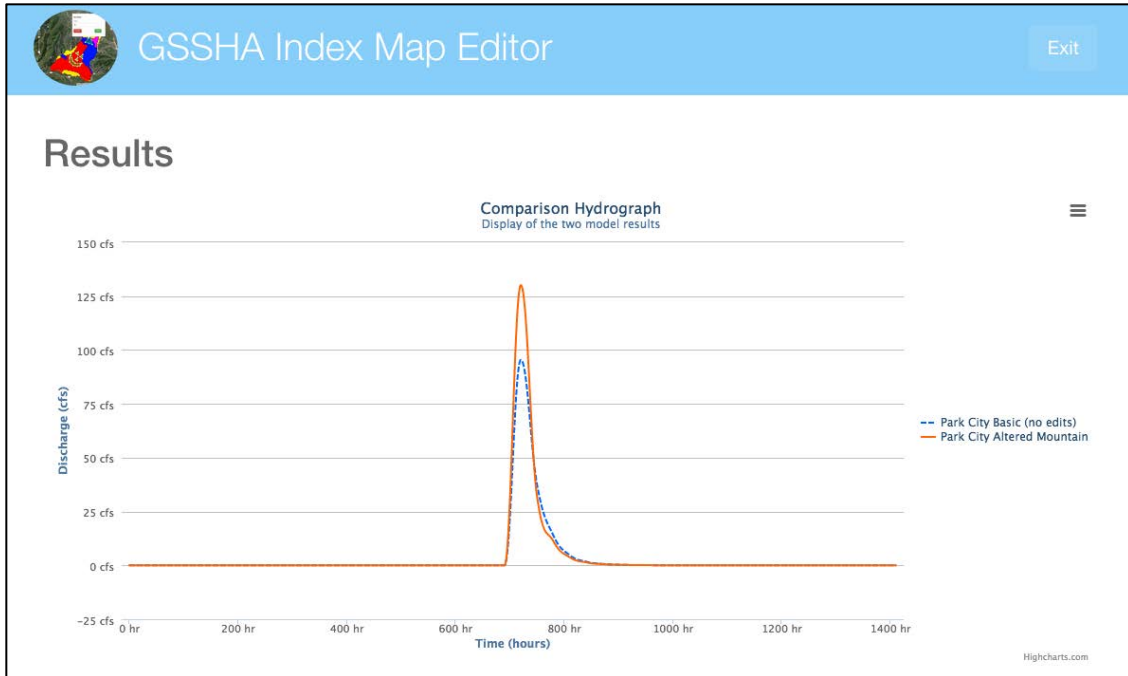


Figure 2-19: Comparison hydrograph for before and after the edits were made.

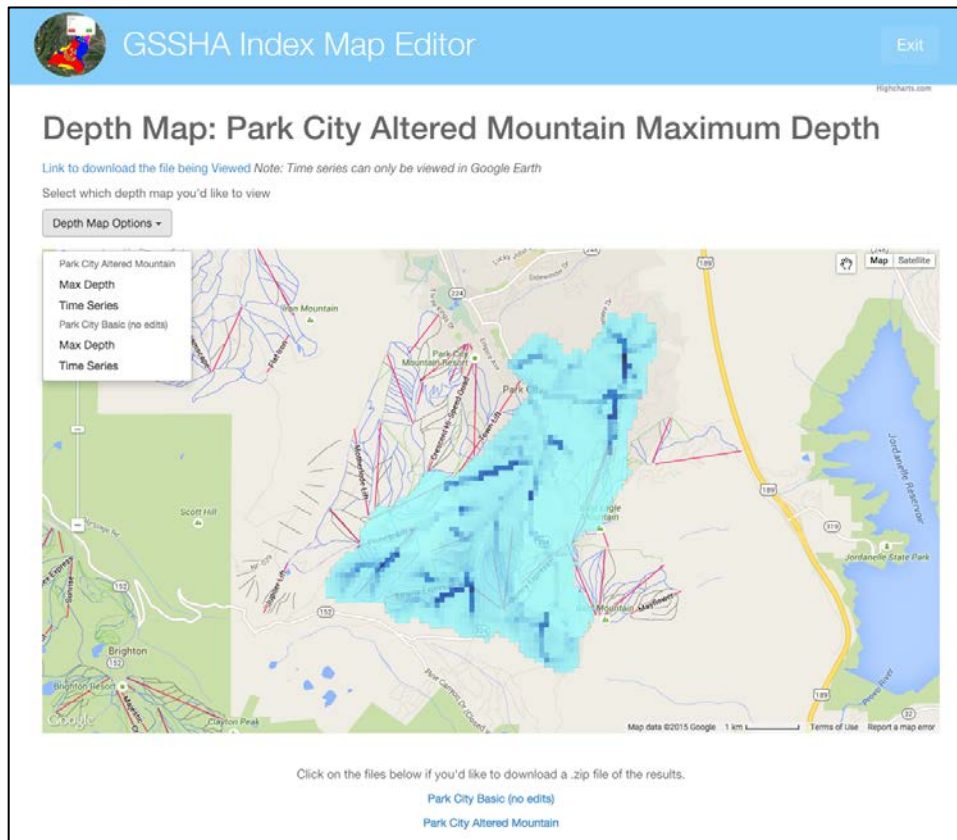


Figure 2-20: Menu with depth map options and depth map display.

## 2.2 Software Organization

In this section I describe the technical details of how the GSSHA Index Map Editor application is implemented. Tethys provides new projects with a folder structure called a “scaffold” when it is installed. This scaffold follows the Model View Controller (MVC) pattern and creates folders to contain controllers, templates, and other information for each application. A basic outline of the file structure can be seen below in Figure 2-21. The controller folder contains Python files that contain the logic to prepare the data for the webpage. The template folder contains the HTML documents for each webpage. The data for the app is stored in a PostgreSQL database and files are temporarily stored in the public folder contained in the scaffold. More information on the Tethys app project structure can be found here: [http://docs.tethys.ci-water.org/en/1.0.0/supplementary/app\\_project.html](http://docs.tethys.ci-water.org/en/1.0.0/supplementary/app_project.html).

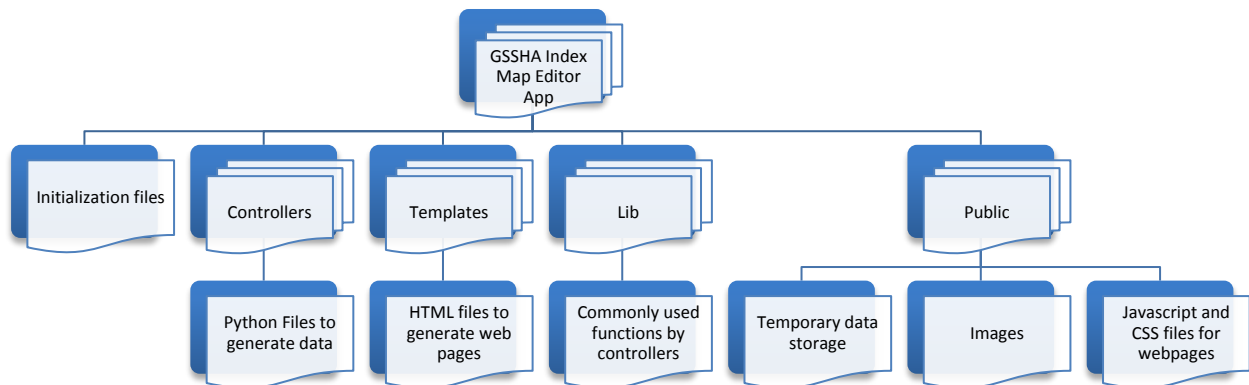


Figure 2-21: Tethys file scaffold applied to new projects and intended contents of files.

## 2.3 Work Flow

In this section I review the general back-end workflow of the app. I will review initialization of the app, how data is stored, and what the app is programmed to do while running.

### 2.3.1 Initialization

When the app is initialized for the first time, two PostgreSQL databases are created. One is formatted using the GsshaPy ORM so it is able to store the GSSHA model data. The other database, referred to as the job database, is prepared to store information regarding each model that undergoes edits and stores things such as links to index map KMLs, a URL link to the original model, the user name of the person making edits, and other pertinent information. In future instances when the app is started, it checks to make sure that both of the databases exist and opens them.

The app also connects to dataset services upon initialization. The GSSHA Index Map Editor app uses a CKAN server and GeoServer, which it connects to at start-up. The connection details are set up by the Tethys Platform administrator when installing the app.

### 2.3.2 Selecting a Model

The controller for the home page uses the Dataset Services API of Tethys to connect to a CKAN database. It automatically queries the database for any zip files that are tagged as GSSHA models. If none are found, a note is displayed for the user to upload one, but otherwise the names are compiled into a list.

When a model is selected from the dropdown menu, the file is downloaded from the CKAN database and stored in a temporary file in the public directory. After the file is unzipped, the mask map file is identified and a GsshaPy MapKit function is called on it to create a KML, which is then stored in CKAN. A URL is returned after the file is uploaded to CKAN and this is stored in the job database along with other information about the model. This URL is then used to display the KML on a Google Map on the webpage. The mask map delineates the watershed boundary and the KML representation of this file is displayed as a preview of the model for the

user. The process uses a caching mechanism so that when the model is selected in the future, the stored URL will be used to display the map and the function to generate a KML will not be run.

### **2.3.3 Loading Model to GsshaPy Database**

One of the novel aspects of the GSSHA Index Map Editor App is its use of an SQL database intermediary for editing the model files. When a model is selected to be edited, a GsshaPy function is called to parse the model files into a GsshaPy database. Once the files are stored in the database, the files can be more easily edited programmatically and the spatial files are exposed to geoprocessing that is available in the form of PostGIS database functions. MapKit uses these database functions to convert spatial files stored in the database to rich KML visualizations. At this point in the app, MapKit is used to convert the index maps to KML. These KMLs are stored in CKAN and their URLs are archived in a dictionary with their respective index name in the job directory. This dictionary is used to populate the HTML page in which users select which index map they would like to edit.

After the files are added to the GsshaPy database, the jobs database is updated to report that the status of the job is “pending”.

All of the available index maps are displayed to the user in a list with a preview map that updates as different index maps are selected. When the “Edit Index Map” button on the HTML page is clicked, a modal appears that gives options for three different methods to edit the selected index map. Depending on the option that is selected, different controllers are called.

### 2.3.4 Editing the Raster Values of Index Maps

After the user selects which method to use in editing the index maps, the corresponding controller is called. The index maps are rasters, so the PostGIS spatial database extender tools are used to edit the values in the maps.

#### 2.3.4.1 Edit by Drawing in Areas

The first controller allows the user to edit index maps by manually drawing in areas and specifying the value to assign to these areas. The editable Google Map gizmo provided through Tethys passes the geometry drawn by the user to the controller in well-known text format and it also passes the value the user supplied for the geometry. If the value passed by the user doesn't exist, it is added to the list of ID values in the database. PostGIS functions are then used to manipulate the geometry into the proper format and to update the database (Figure 2-22). The `ST_GeomFromText` function is used to change the well-known text to the necessary geometry format. The geometry is then transformed to the projection required for Google Maps. The `ST_SetValue` function is then used to update the index raster file in the database with the new geometry and the specified value. Finally, if there were any index values that are no longer present after the new geometry was added, those values are removed from the list of ID values in the database.

```
# Change values in the index map
change_index_values = "SELECT ST_SetValue(raster,1, ST_Transform(ST_GeomFromText('{0}', 4326),{1}),{2}) " \
"FROM idx_index_maps " \
"WHERE id = {3};".format(wkt, srid, value, index_raster.id)
```

Figure 2-22: Sample code used to change the index map values based on the geometry provided.

#### 2.3.4.2 Edit by Shapefile

The second controller option allows users to upload a shapefile and use it to replace values in the index map. The HTML on this page has a form that passes on chosen files to a controller. These files must include a “.shx”, “.shp”, “.dbf”, and “.prj” file. These files are

uploaded to a temporary public folder, zipped, and uploaded to GeoServer. A “WMS” layer is retrieved from GeoServer and displayed on a Google Map on the upload page. If the user accepts the uploaded shapefile as the file they would like to use, a controller is called to retrieve GeoJSON from GeoServer and use it to replace the index map raster values. It follows a similar pattern as the replacement function used when editing the index with well-known text geometry, but instead of using ST\_GeomFromText, the function ST\_GeomFromGeoJSON is used, as seen in Figure 2-23. After all the values are replaced, if values are added that didn’t exist, they are added to the ID list in the database and if they are no longer present, they are removed.

```
# Change values in the index
change_index_values = "SELECT ST_SetValue(raster,1,ST_Transform(ST_GeomFromGeoJSON('{0}'), {1}},{2}) " \
"FROM idx_index_maps " \
"WHERE id = {3};".format(str(geom_full), project_file_srid, id, index_raster.id)
```

**Figure 2-23: Sample code used to change index map values based on the GeoJSON provided.**

### 2.3.4.3 Edit by Combining Index Maps

The third controller allows users to replace an index raster with an existing raster or combine two rasters. If only one index map is selected to replace the selected index map, an “UPDATE” query is used on the database to update the index raster with the raster from the selected index map. The ID values are also updated to match those in the new raster.

When two index maps are selected to replace the selected index map, an “UPDATE” query is run to merge the two index maps and give unique values. These unique values are used to update the ID values for the index map in the database.

### 2.3.5 Updating Index Variable Values

After the raster for the index map has been edited, the values for the associated variables for each ID value can also be changed. The HTML page following updating the index map raster



values includes a form with a table of the GSSHA mapping table values associated with the index map that can be updated. When the “Next” button is clicked, all the values in the form are used to update the mapping table values in the database.

### **2.3.6 Submitting the Altered Model**

After the user edits the index map or maps, the “Submit New GSSHA File for Comparison” button is pressed and a modal with a form allows the user to input a name and description for the new model. Both the name and the description are stored in the job database as well as the status being changed to “submitted”. If there is not a GFL file for the project, that project card is added to the model in the GsshaPy database. The input files for the edited model are then extracted from the database using GsshaPy functions and saved as a zip file. The file is stored in a temporary file in the public directory and then sent to be stored in CKAN.

### **2.3.7 Viewing Edited and Submitted Models**

There are two HTML pages that display models that have been altered. One page displays models that have been edited, but have not yet been submitted. The list of models is compiled by querying the job database for all of the models that are currently being edited, or have a status of “pending” in the job database. There is a button with each of these models that calls the controller which returns HTML displaying the model opening page. There is also a button that runs a query to delete the model from the database.

The other page displays the models that have been submitted by querying the job database to find models that have been submitted, or have a status of “ready to run” in the job database. There are several controls that can be called from this HTML page. One controller

deletes models from the job database, one runs the models, and the last one prepares the results page.

## 2.4 Deleting Models from the Database

The controller to delete models from the job database takes the model ID, uses a query to get a handle on the model using the ID, and then deletes it from the database. The model information still remains in the GsshaPy database, but it is removed from the job database and will no longer be accessible to the user.

## 2.5 Running Models

A web processing service (WPS) is used to run the models. The 52 North WPS component of Tethys Platform is used to execute the models from a publicly accessible URL and it returns a zipped file of the results.

The first model to be run is the one that has been altered. The CKAN URL where the zipped file is located is submitted to the WPS. The WPS downloads the zipped file, runs the GSSHA model, and returns a zipped file of the results. The results are then added to CKAN with a label of “Certified”. The label “Certified” implies that it has all of the results files needed for it to be displayed on the results page. The results files that are needed include an OTL and GFL file.

The next model to be run is the original model. If it had a label of “Certified”, the model is not run, but the URL that is located in the job database is passed on. But if the model is not certified, it needs to be run.

Before the model is run, a check is run to see if there is a GFL file present. This file is required to produce a maximum depth map. If it is present, the model is sent to the WPS just like

the altered model was. Otherwise the model is uploaded to the GsshaPy database and a GFL project card is added. The file is then extracted from the database, zipped, stored to CKAN, and run as the altered model was run.

While the models are run, the status for the job in the job database is changed to “processing” and this is displayed on the HTML page. But once the models are completed running, the status is changed to “complete” and the page is refreshed. The HTML page displays a button for the view results controller for models with the status “complete”.

## 2.6 Viewing Model Results

When the controller is called to view the model results, the original and altered model result files are downloaded from CKAN. The OTL files are used in combination with a HighCharts gizmo in Tethys to produce a comparison hydrograph.

A Google Map is included on the page to display PNG images of the maximum depth maps and time series depth maps of the new and original models. When an option is selected from the drop-down menu, the PNG image is produced, displayed, and a link is provided for the user to download the image.

A link to the zipped result files on CKAN is provided and clicking on it will download the file to the users computer.

### **3 RESULTS**

An experimental case study was used to illustrate how the app would be used by a municipality facing land use changes. It was based off a GSSHA model prepared for Park City, Utah.

#### **3.1 Introduction to the Test Case**

Park City is located in the northern half of Utah in the Wasatch Mountain Range. It is the home of three major ski resorts and has an expanding tourism economy. The central part of the city is below a small watershed formed by the Wasatch Mountains. Because most of the city is immediately below the watershed, it is important for Park City to know what kinds of flows to expect from the mountains after storms or during spring runoff. To help in flood and runoff prediction, a GSSHA model has been prepared to simulate a storm that they could expect and would need to plan for. In this test case we will assume that a developer is desiring to change an area of the mountainside to apartments and a ski resort, and that the city is analyzing the effect of the land use changes.

#### **3.2 GSSHA Model Assumptions**

The GSSHA model covers the watershed above Park City, covering an area of 7.3 square miles. The storm type is a Modified Type II 24-hour storm with a total accumulation of 2.5

inches of rainfall. The rainfall is broken up into 6-minute increments. There are three index maps associated with this model: soil, land use, and a combination map. The variable that relates to the land use mapping table is roughness and soil moisture is specified through the soil mapping table. The combination mapping table controls seven variables: hydraulic conductivity, capillary head, porosity, pore index, residual saturation, field capacity, and wilting point.

### 3.3 Test Case Analysis

In the first instance, we will assume that a developer comes in to meet with the city and manually draws in a proposed area of development on the eastern side of the watershed. In the second instance, we will assume that a developer sends the city a shapefile of the location of the proposed development for the city to use. In both cases, the city is interested in the change of runoff in order to plan for changes in storm water flows. The process of preparing both test cases will be discussed in the following sections.

### 3.4 Test Case 1

A polygon was drawn in on the eastern side of the watershed by hand, as shown in Figure 3-1. A lower roughness value was expected after the forest was removed and the impervious area was increased, so a roughness value of 0.01 was assigned, as shown in Figure 3-2. After the changes were made, the model was submitted and run.

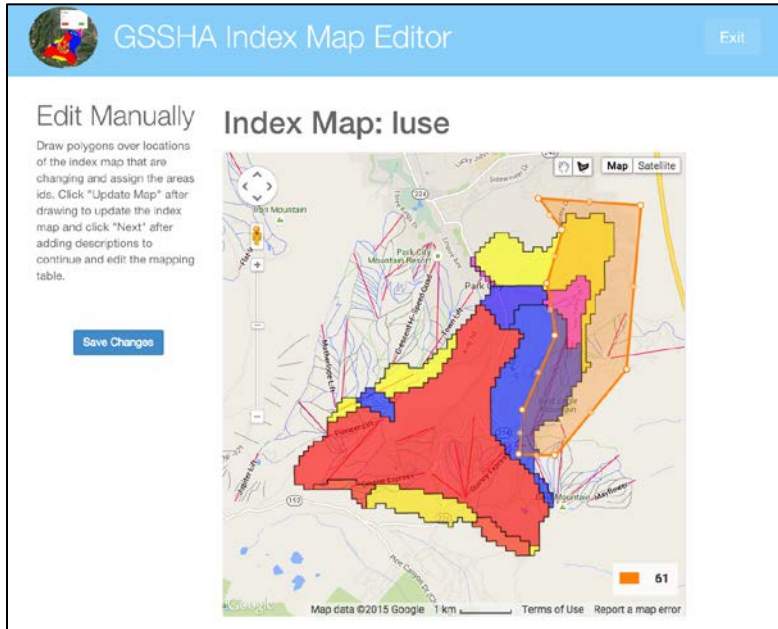


Figure 3-1: Test case 1 – Development on eastern side of the watershed.

### Edit Index Mapping Table Values

Below is a list of mapping tables you can edit that are related to this index map. After making changes to the values, click "Next" to continue and review the mapping tables.

**ROUGHNESS**

### Index Map: luse

ID	Description 1	Description 2	ROUGH
11	Untitled landuse		<input type="text" value="0.011"/>
16	Untitled landuse		<input type="text" value="0.011"/>
17	Untitled landuse		<input type="text" value="0.011"/>
32	Untitled landuse		<input type="text" value="0.05"/>
33	Untitled landuse		<input type="text" value="0.04"/>
41	Untitled landuse		<input type="text" value="0.1"/>
42	Untitled landuse		<input type="text" value="0.15"/>
61	Development		<input type="text" value="0.01"/>

Figure 3-2: Input roughness values for Test Case 1.

### 3.4.1 Results of Test Case 1

The resulting hydrograph of Test Case 1 is shown in Figure 3-3. The peak flow on the hydrograph without the edits was 95.41 cfs and with the edits it was 101.73 cfs. Maximum and time series depth maps were also produced and were displayed and were available for download. The maximum depth maps can be seen in Figure 3-4. These maps show a slightly lower maximum depth on the eastern side of the watershed. The higher peak on the hydrograph and lower maximum depth are to be expected as there is less roughness, so the water will not get captured and will run off more quickly.

Results for the test case were retrieved in a little over 30 minutes after the models were submitted. The first GSSHA model completed running in 10.6 seconds and the second model completed running in 20 minutes. The rest of the time was used in preparing the original model to be run by reading it into the database, adding the GFL card, and reading it back out to a zipped file. This model was run during a time when many other models were in the web processing service queue. With additional computing resources the time required to run the models decreases significantly. This model was run a second time when the queue was smaller and it only took 22 minutes to retrieve results.

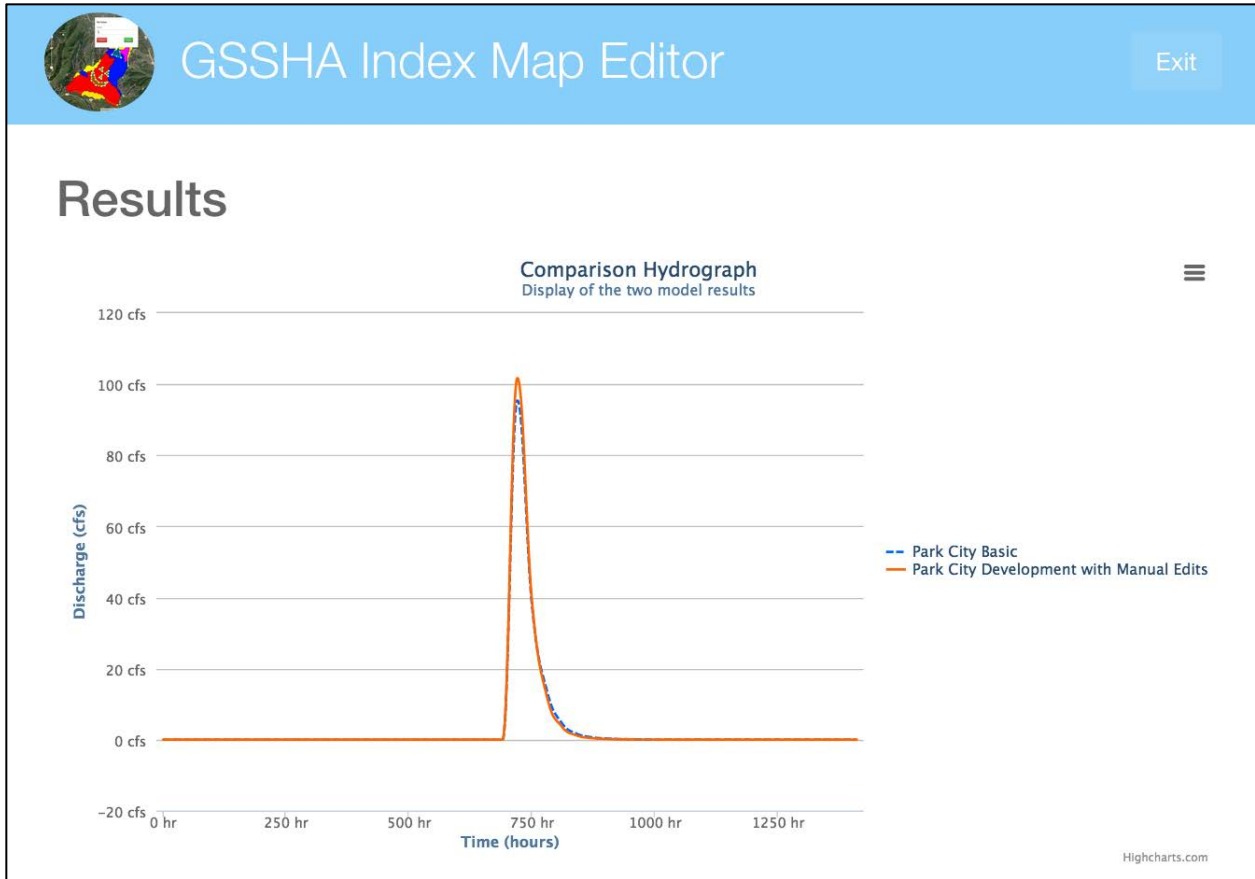


Figure 3-3: Comparison hydrograph for Test Case 1.

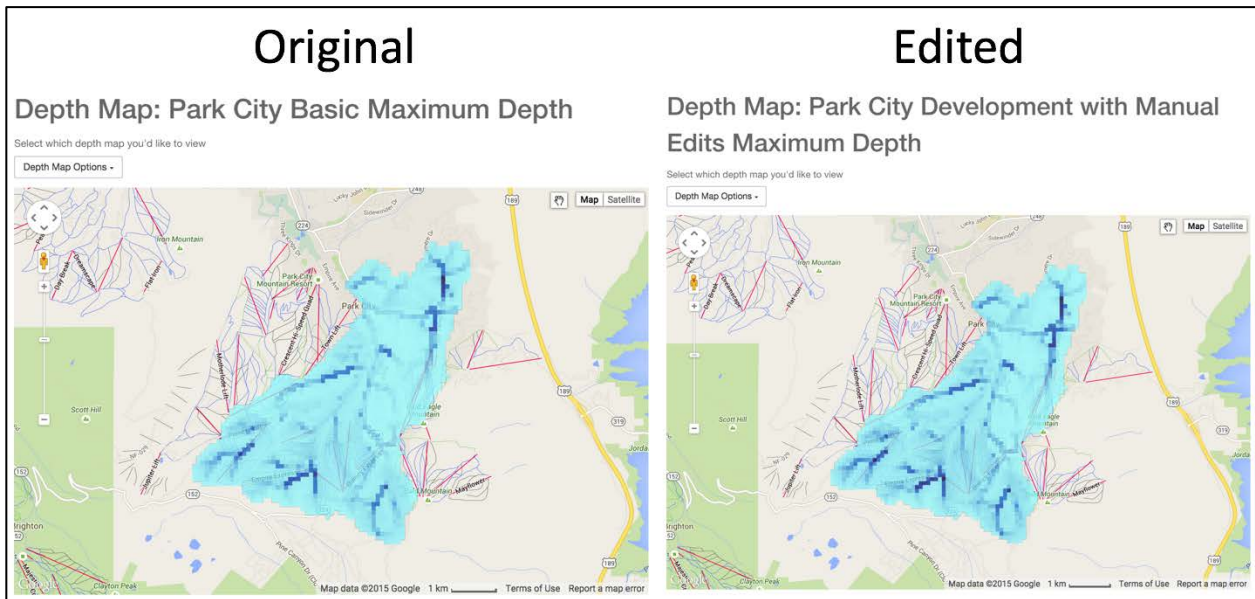


Figure 3-4: Resulting maximum depth maps for Test Case 1 showing less maximum depth on the eastern side of the watershed after the roughness was lowered.



### 3.5 Test Case 2

A shapefile was created to show possible development locations in the Park City watershed and was uploaded on the app (see Figure 3-5). After the shapefile was uploaded, it was merged with the land use index map to produce the second image shown in Figure 3-6. In this image the development areas are displayed in red. As was done in Test Case 1, a roughness value of 0.01 was given to these areas. After these changes were made the models were submitted and run.

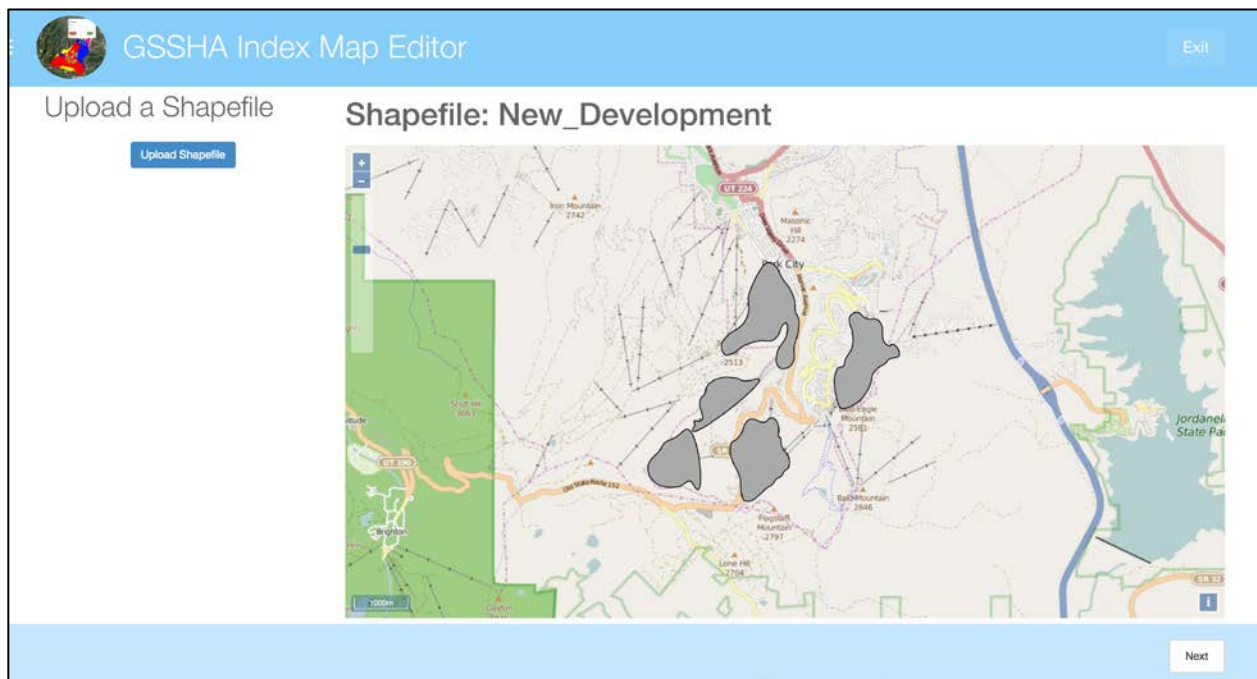
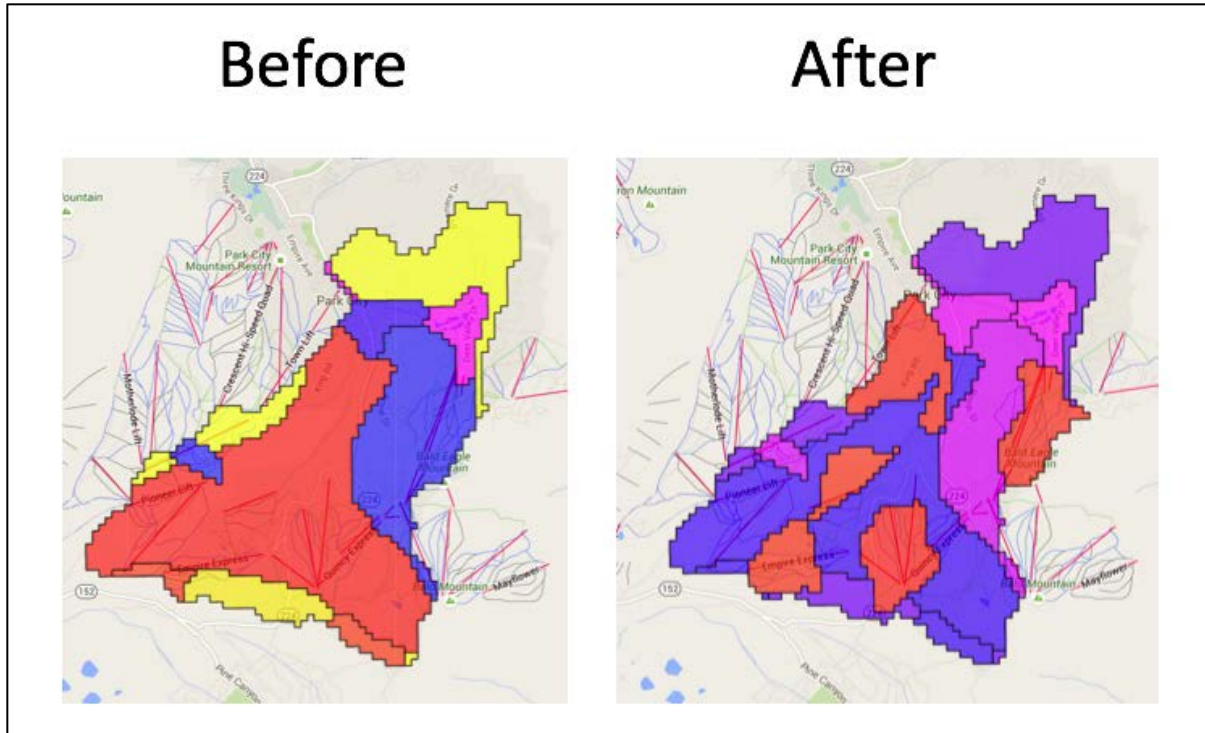


Figure 3-5: Test Case 2 shapefiles showing development.



**Figure 3-6: Test Case 2 – Land use index map before and after updating by shapefile.**

### 3.5.1 Results of Test Case 2

The resulting hydrograph of Test Case 2 is shown in Figure 3-7. The peak flow of the original model was 95.41 cfs and the peak flow of the edited model was 116.37 cfs. Maximum and time series depth maps were also produced and were displayed and were available for download. The maximum depth maps can be seen in Figure 3-8. The maximum depth in the areas of development is lower than it was previously. The lower roughness in these areas prevents capture of the water, leading to a lower maximum depth and a higher peak flow at the watershed outlet.

The results for this test case were returned 3 minutes after the models were submitted. The first model took 1 minute and 13 seconds to run and the second model took 1 minute and 28 seconds to run. The rest of the time was used in preparing the original model to be run by reading it into the database, adding the GFL card, and reading it back out to a zipped file. This test case

did not take very long to run because the web processing queue was relatively small and there were sufficient computing resources available. When this model was run a second time when the queue was larger it took 18 minutes to retrieve results after it was submitted.

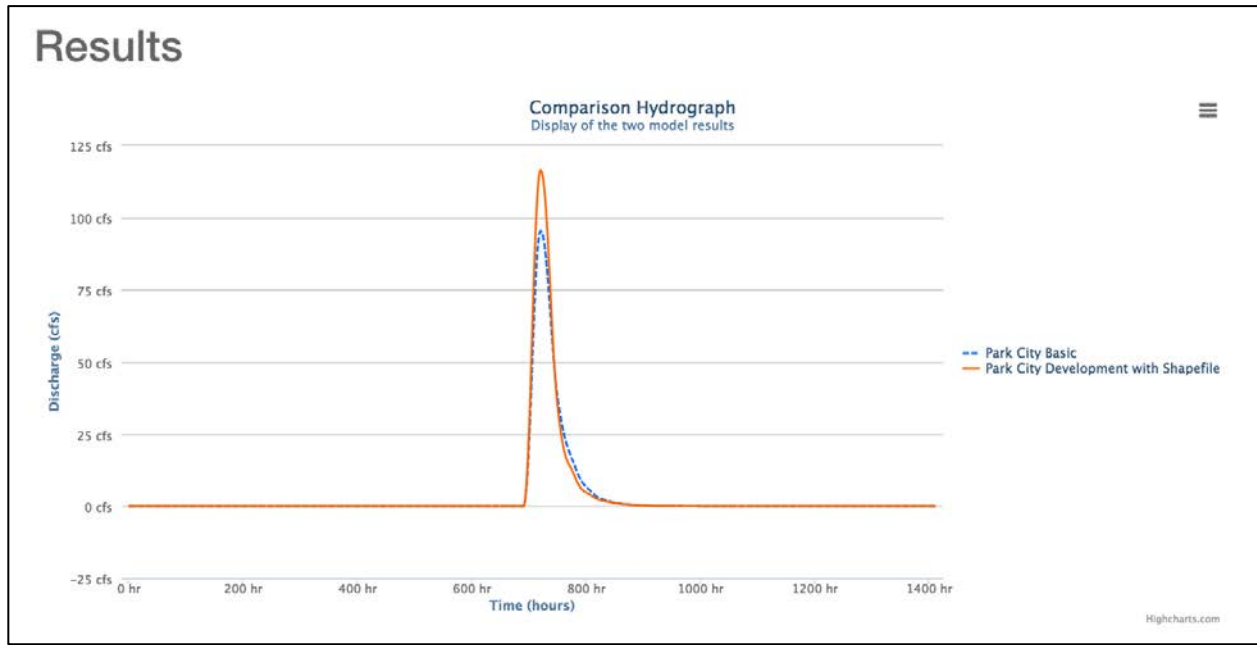


Figure 3-7: Comparison hydrograph for Test Case 2.

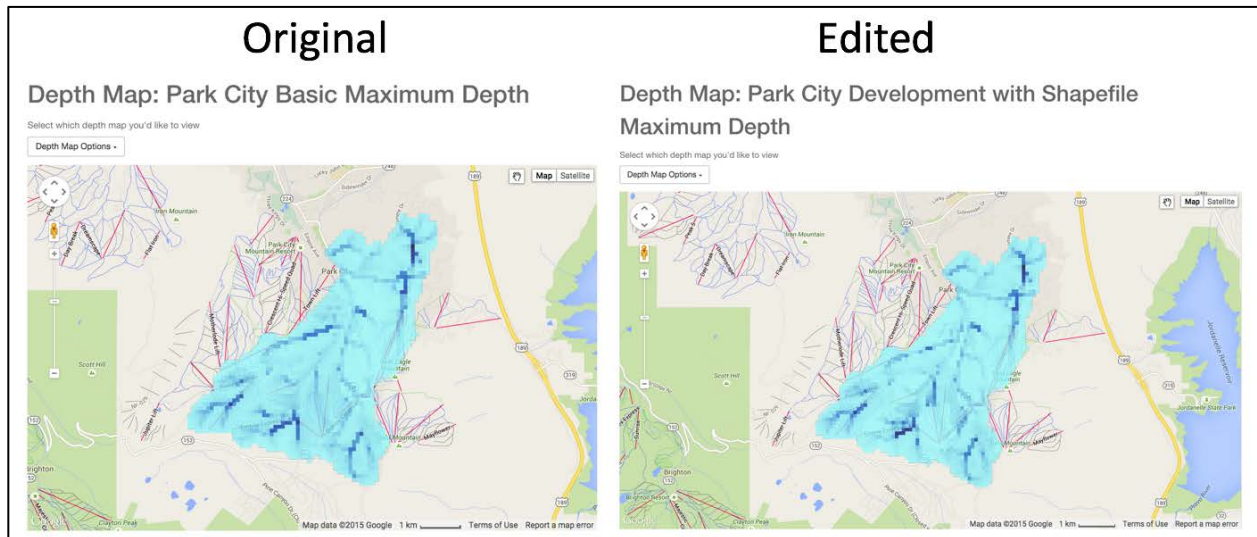


Figure 3-8: Resulting maximum depth maps for Test Case 2 showing a lower maximum depth in the areas of development.

### 3.6 Overview of Results

This tool provides a method for individuals with no knowledge about GSSHA input files to easily make edits to models and then view the results all in a web browser. With information provided from these model results, Park City officials could make critical storm water management decisions. Knowing exactly how much extra storm water would be produced could aid in decisions on whether detention would be required to control outflow from the area or if any other measures needed to be taken to protect downtown from runoff from the watershed.

## 4 CONCLUSIONS

The GSSHA Index Map Editor app provides a simple and quick way for users to alter land use in GSSHA models and retrieve results of the new model to compare with the original. It is a powerful tool that can be accessed anywhere there is Internet access and produces accurate results. Unlike some other web-based modeling tools, it provides a way to edit and run spatially distributed GSSHA models quickly and easily.

### 4.1 Review of Objectives

The research objectives outlined in the introduction are reevaluated here in light of the research accomplishments.

#### 4.1.1 Simple Mechanism for Editing Index Maps

The first objective was to provide a relatively simple mechanism for editing index maps. This was accomplished by providing three simple methods for users to make changes to index maps including manually drawing in edits, uploading a shapefile, or combining index maps to replace the selected index map. There are a few straightforward steps to accomplish each method followed by a table where the user makes edits to the values of the index map. All of the changes are stored in a PostgreSQL database and after edits are made, the model can be written out to a zipped file from the database.

#### **4.1.2 Integrate Index Map Editor into Web-Based Modeling System**

The second objective was to integrate an index map editor into a web-based modeling system to allow the application to be used across various platforms. The Tethys Platform was used because it is a web based application and it provides the ability to link to a PostgreSQL database. Tethys gizmos and PostGIS tools were used to implement the methods of editing the index maps. Using PostGIS and GsshaPy, the information for the model and the updated index maps is stored on a PostgreSQL database that is hosted on the same server as the Tethys Platform.

#### **4.1.3 Demonstrate How the System Can Be Implemented**

The third objective was to demonstrate how this system can be implemented in a web-based GIS decision support system to be used by decision makers. The system was designed so that it would be simple to use and user friendly. It only allows users to change certain aspects of the model, preventing damaging changes from being made.

There are a number of projected users of the application including city managers and planners, organizations such as the forest service, and students at universities. As city managers and planners are evaluating flooding and runoff problems that could affect their infrastructure, this app would provide the ability to quickly and easily identify threats as land use changes. Cities can hire a qualified engineering firm to prepare a highly sophisticated GSSHA model that they can then use in the GSSHA Index Map Editor app. Then, as cities develop and as impervious areas increase, city managers can know what improvements need to be made in terms of storm water management. Cities could know exactly how much water needs to be detained by developers and have justification in requiring them to do so. Cities could also use model results to plan sustainable long-term infrastructure improvements.

Other organizations could also benefit from using this application. One such organization is the Forest Service that could use it following forest fires. After a fire, the ground may be hydrophobic in the affected areas. It would be useful at this point for the forest service to be able predict potential flooding problems by altering a model to match the new existing characteristics of the burned area. This app allows them to do so and then run the model and view the resulting differences in runoff. They would quickly know how their watershed would be affected by the sudden change.

Students would also benefit from using this application, especially where modeling software licenses are not available. This application provides a method for students to visualize the effect of land use changes and to make edits to models without needing a software package on their computer.

#### **4.1.4 Explore the Capability of Tethys in Meeting Needs**

The fourth objective was to explore the capabilities of Tethys for meeting the beforementioned needs. Tethys provided an excellent platform for development of hydrologic modelling applications. Tethys provided tools and packages that allowed the GSSHA Index Map Editor app to quickly be produced. In the following sections I will discuss the advantages and limitations of Tethys for hydrologic modeling applications.

The release package that contains the app package makes it possible for the app programmer to focus on the logic of the app itself rather than needing to program the background for where the app will be hosted. The gizmos that are supplied with Tethys allow programmers to easily provide maps, date pickers, toggles, tables, and many more features on their web page with a simple gizmo reference and input information.

Tethys provides other features that are critical in hydrologic apps. One feature includes the simple methods for setting up data storage. The ability to connect to data storage such as CKAN and HydroShare provides access to large amounts of data storage that isn't possible otherwise. Also, the ability to quickly set up PostgreSQL databases with PostGIS extensions makes it simple to store and manipulate spatial data. Finally, the connections to geoprocessing through services such as GeoServer provides function that is needed in many hydrologic apps.

There are some areas in which Tethys could be improved for hydrologic apps. One area where Tethys is lacking is that it is not possible to view time series maps. Google Earth is not supported with Tethys and there is no way provided to view time series spatial data.

#### **4.1.5 Test the Developed System in a Case Study**

The fifth objective was to test the developed system using a case study. The GSSHA Index Map Editor App was used for a case study of possible land use changes in Park City, Utah. Making land use changes was simple and fast. The results took longer to produce when the server was busy, but otherwise the models ran as quickly as they would on a desktop application. The results were displayed in a manner that made the before-and-after scenarios easy to compare and results were also available for download.

#### **4.1.6 Recommendations Regarding Use of Tethys for Similar Modeling Applications**

The last objective was to make recommendations regarding the application of web-based GIS tools, specifically Tethys, for future development of similar modeling applications. Overall, Tethys is one of the best possible solutions for building a hydrologic web application for someone with little programming experience. It allows users to program their controllers in



Python, a common programming language in the sciences, and only requires them to understand a little HTML.

Tethys is still under development and changes are rapidly made. This is a benefit as this means that new features are being released with new versions and the functionality of Tethys is still growing. New gizmos are released with each version making building applications easier for developers with less web programming experience.

The continuing development of Tethys also poses a challenge. During development of this application, features changed several times and the code had to be updated. Generally Tethys developers ensure that updates are backwards compatible, but developers should be aware that their applications may need to be updated as Tethys continues to develop.

#### 4.2 Opportunities for Continued Development

There are several ways in which the GSSHA Index Map Editor app could be improved. One of these includes the method in which models are run. Models are currently sent through a WPS to 52 North, but they quickly maximize the available computing power. Another computing method that may provide the needed additional computing power would be HTCondor. HTCondor uses idle computing power that is available and farms job to the available resources. This would prevent a backlog of model runs and allow results to be returned to the user more quickly.

Another feature that could be a good addition to the app would be allowing the user to select what kind of output results they would like. The app produces a hydrograph, maximum depth map, and time series depth map for each model, but additional information could include things such as a stream depth map. The app could benefit from user customization of the results produced.

One feature of the app that is in need of some continued development is the replace by shapefile feature. If the shapefile contains a large amount of features, the database times out before the index map is updated. This may be able to be solved by clipping the shapefile by the mask map boundaries before running the “Update” function on the database.

#### 4.3 Software Availability

The source code for the GSSHA Index Map Editor App is available to view and download at <https://github.com/CI-WATER/tethysapp-gsshaindex>. A live demo of the app is available at <http://demo.tethys.ci-water.org/apps/gsshaindex>.

## REFERENCES

- Choi, J., Bernard Engel, and Richard Farnsworth. 2005. "Web-based GIS and spatial decision support system for watershed management." *Journal of Hydroinformatics* 7: 165-174.
- CI-WATER. 2015. "CI-WATER: A Utah-Wyoming Cyberinfrastructure Water Modeling Collaboration." (accessed 5/9, 2015). <http://ci-water.org/>.
- Downer, Charles W., EJ Nelson, and Aaron Byrd. 2003. *Primer: Using Watershed Modeling System (WMS) for Gridded Surface Subsurface Hydrologic Analysis (GSSHA) Data Development- WMS 6.1 and GSSHA 1.43 C*.
- Downer, Charles W. and Fred L. Ogden. 2006. *Gridded Surface Subsurface Hydrologic Analysis (GSSHA) User's Manual; Version 1.43 for Watershed Modeling System 6.1*.
- Downer, Charles W. and Fred L. Ogden. 2004. "GSSHA: Model to simulate diverse stream flow producing processes." *Journal of Hydrologic Engineering* 9, no. 3: 161-174.
- PostGIS Developers. 2015. "PostGIS: Spatial and Geographic objects for PostgreSQL." (accessed 5/7, 2015). <http://postgis.net/>.
- Swain, Nathan. 2014. "Gsshapy Documentation." (accessed 5/5, 2015). <http://gsshapy.readthedocs.org/en/latest/index.html>.
- Swain, Nathan. 2014. "Tethys Platform 1.0." (accessed 3/14, 2015). <http://docs.tethys.ci-water.org/en/latest/index.html>.
- Wilson, John P., Helena Mitsova, and Dawn J. Wright. 2000. "Water resource applications of geographic information systems." *Urisa Journal* 12, no. 2: 61-79.