Feasibility of Using Existing Web Services for On-Demand Data Access within Distributed Environmental Decision Support Systems

A Thesis

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

in partial fulfillment

of the requirements for the degree

Master of Science

By

Stuart F. Sheffield

August

2016

APPROVAL SHEET

The thesis

is submitted in partial fulfillment of the requirements

for the degree of

Master of Science

Stuart F. Sheffield

Author

The thesis has been read and approved by the examining committee:

Jonathan L. Goodall, Ph.D., P.E.

Advisor

Teresa B. Culver, Ph.D.

John H. Porter, Ph.D.

G. Michael Fitch, Ph.D.

Accepted for the School of Engineering and Applied Science:

[illegible]

Craig H. Benson, Dean, School of Engineering and Applied Science

August

2016

ABSTRACT

Environmental decision support systems (DSSs) are becoming more data-intensive as data availability increases. Significant effort has been invested in making these data accessible through web services, which allow for on-demand data access through standardized, machine-accessible interfaces. Although the typical approach is to use web services to gather and store data as a separate, off-line step, there are advantages to being able to build distributed environmental DSSs where data are accessed on-demand from web services for direct use within the DSS. The objective of this study is to assess the current feasibility of this approach by testing the response times of existing, national-scale web services in the United States that would be commonly used within a distributed environmental DSS. A stormwater management application was used to define typical data needs for an environmental DSS, these data needs were mapped to available national-scale web services, and then the service response times were measured under different scenarios. Results of the analysis showed that service requests, which included soil data access, river network tracing, watershed delineation, and streamflow statistics, took from less than a second to more than one minute to respond. Some services scaled well to increasing study area sizes, while others did not. All services showed significant variability depending on the geographic location of the client. These results highlight specific areas where future research and development are needed to improve service response times in order to achieve the vision of distributed environmental DSSs.

Acknowledgements

I am appreciative of all aid I received from others during my time at the University of Virginia. Especially from my advisor, Jonathan Goodall, whose assistance and guidance were essential to not only my research but also to my interest in web technologies. Additionally, I would like to thank the members of my thesis committee, Dr. Teresa Culver, Dr. John Porter, and Dr. Michael Fitch, for their time. I would also like to think Mr. Roy Mills and Mr. John Olenik at VDOT for their assistance. A final thank you to friends, both within and outside of my research group, who made my time at UVA fly by, and to my family whose support has always been there.

Table of Contents

List of Figures

List of Tables

INTRODUCTION

The advent of the Internet has given researchers, scientists, and engineers the ability to quickly access and use data from different sources. The volume of data being produced in scientific fields is doubling yearly (Szalay and Gray, 2006) and these data are increasingly being placed in online repositories and databases. Technologies like sensor networks and remote sensing have contributed to the surge of data with their increased use across scientific fields and generation of large, high resolution datasets (Philip Chen and Zhang, 2014). With data availability growing, the need for tools that can automatically access information from different sources, then process and analyze the data to provide meaningful results for environmental management, has become essential (Hey and Trefethen, 2005; Tarboton et al., 2014).

Web services have become a popular method for automating access to scientific datasets. Web services are designed to communicate messages in a standardized format between computers over the Internet, allowing geographically separated computers to easily transfer data. Mineter et al. (2003) foresaw the need for the new generation of environmental applications to shift away from the desktop computer and toward more distributed resources interconnected through web services. Although web services are being employed for data access, they are also progressively being used to produce derived data through more advanced analysis, visualization, and modeling performed on-demand based on user requests. Web service approaches have been proposed for various aspects of environmental applications including data analysis, visualization, and model simulation (Booth et al., 2011; Díaz et al., 2008; Feng et al., 2011; Goodall et al., 2011; Granell et al., 2010; Walker and Chapra, 2014).

In recent years, several major federal agencies have begun to offer web services to access data stored on their servers. Examples include the Environmental Protection Agency’s (EPA) Watershed Assessment, Tracking and Environmental Results System (WATERS) and the Storage and Retrieval and Water Quality Exchange (STORET), the United States Geological Survey’s (USGS) National Water Information System (NWIS) and StreamStats application, and the United States Department of Agriculture’s (USDA) Soils Data Access. The Consortium of Universities for the Advancement of Hydrologic Science, Inc. (CUAHSI) Hydrologic Information System (HIS) project proposed web service standards to improve hydrologic time series access, as well as software for both server and client side data management within a distributed HIS (Ames et al., 2012; Goodall et al., 2008; Horsburgh et al., 2010, 2009; Tarboton et al., 2014). Standardization of web services makes automated data access from heterogeneous data sources easier by providing a common interface for communicating between clients and servers.

The increased availability of standardized web services suggests that future Decision Support Systems (DSS) will be able to leverage common data, analysis, simulation, and visualization resources on-demand to support decision makers (Buytaert et al., 2012; Laniak et al., 2013). Although web services are commonly used now for data access within environmental DSSs as part of a preliminary, off-line data gathering step, there are significant advantages to having a distributed system where the services are used to integrate data, on-demand. One key advantage of the data or calculations being offered through a web service is that erroneous services can be changed and all clients will have access to the corrected information without the need to install new client-side software (Buytaert et al., 2012; Goodall et al., 2011). This ability also raises a concern that reproducing past studies may be compromised due to unanticipated changes in underlying services. To address this concern, there would need to be clear and consistent ways to maintain versions of services and to alter users to updated services. Clever ways for archiving analyses including the data and models that can be used to reproduce the analysis will also be important. While admittedly more complex when dealing with distributed, service-oriented systems these challenges of versioning, computational reproducibility, and provenance exist whether using a distributed or centralized DSS architecture.

Much of the prior research toward this vision has been directed at designing DSS web services themselves or evaluating architectures and their suitability for an environmental DSS (Matthies et al., 2007; Sun, 2013). There has been less work examining the performance of available web services for supporting a distributed environmental DSS. Therefore, the objective of this research is to assess the feasibility of building a distributed environmental DSS using existing, authoritative, national-scale web services in the United States. This objective is explored through a stormwater management application used to identify data needs and map those data needs to available web services. A series of experiments were conducted to measure the response times of the service requests for different data access needs. The primary contribution of this research is to better understand the current state of web services for creating a distributed environmental DSS, and to identify potential bottlenecks where future research and development could be directed in order to speed up web services and move toward the longer term vision of distributed environmental DSSs.

The remainder of this thesis is organized as follows. The background section introduces web services and their potential role in an environmental DSS as well as an environmental management application used to design a set of experiments for testing the feasibility of an environmental DSS. The methodology section describes the web services used for the analysis as well as the specific experiments used to test the services. This is followed by a results and discussion section where results from the individual experiments are presented and discussed in terms of performance, reliability, and variability. Finally, conclusions from the study findings are presented, along with suggestions for future research building from this study.

BACKGROUND

Web Services

Web Services are defined as software systems designed to support machine-to-machine interaction over a network. They are a way to implement service-oriented architecture (SOA), which is an architectural pattern in which application components provide services to other components via a network (Microsoft, 2016; The Open Group, 2016; Barry, 2016). Services, in this case, refer to the implementation of a well-defined function. The services contained within a SOA application are reusable, flexible, and independent from one another.

There are two roles defined in a web service: a service provider and a service consumer or client. The service provider creates the web service, publishes access information, and registers what is available to the client. The client must find the web services and invoke them to access available information or features. Two common methods to implement web services are Simple Object Access Protocol (SOAP) and Representational State Transfer (REST). SOAP is a protocol or standard for exchanging structured information while REST is an architectural style. Applications that employ REST principles are called RESTful and were used for the web services tested in this study. To be called RESTful, applications must satisfy six constraints defined by Fielding (2000). These constraints include a separation of client and server, a lack of client storage on the server between requests, and a uniform interface.

RESTful services can be accessed by the client much the same way that internet browsers load web pages. Resources, such as information and data, are requested through a Uniform Resource Identifier which can be contained in a URL. Additional options such as response formats and search parameters would also be contained in the URL. A general workflow for web services involves the client first invoking the web service by sending a message or request to the provider over the network using a URL with all necessary identification information. The provider reads the message, obtains the requested information, and sends a message back to the client containing the requested information over the network.

Stormwater Management Scenario

Virginia Stormwater Management Program (VSMP) regulations require construction projects to account for stormwater runoff impacts from increased impervious surfaces in order to prevent erosion, flooding, and water quality reduction. Organizations have traditionally constructed onsite stormwater Best Management Practices (BMPs) to mitigate these stormwater impacts. Recently, changes to the regulations allow for nutrient credit purchases as an alternative to onsite BMP construction. This new option allows pollutant dischargers to purchase credits from off-site sources to offset what would be treated onsite.

To qualify for the use of nutrient credits, a project must meet one of the following criteria determined by the Code of Virginia § 62.1-44.15-35. First, the project area must contain less than 5 acres of disturbed land. Second, the post-construction phosphorus control requirement must be less than 10 pounds per year. Third, if the first two criteria are not met and if the applicant can demonstrate onsite control of at least 75%, the remaining required reductions can be met through the purchase of nutrient credits. Project managers should account for other considerations. If the project discharges into a local watershed with an established nutrient TMDL, nutrient credits may still be purchased provided that the use of the credits do not prevent compliance with the local limitation.

There are several calculations needed to check whether or not a project meets the eligibility criteria. A key calculation is the determination of the amount of phosphorus generated by the site. Traditionally, this is done using the procedure described in the Virginia Runoff Reduction Method (VRRM) using general site information, such as total acreage, soil types, land cover, and BMP types, and a regression equation to estimate phosphorus runoff. An alternative method is to use a surface water model, such as TR-20, to estimate runoff and pollutant loads. The model requires the drainage area, curve numbers, and concentration times which can be calculated from land use, soil, and elevation data for the watershed. This stormwater management application is ideal for the purposes of testing how well web services perform because it is simple, but still requires a broad set of input data that is fairly typical of other environmental applications.

Data Description

A summary of the data needed to support nutrient trading eligibility is provided in Table 1. It is assumed that the user will provide some of the inputs to the application, including the coordinates of polygon vertices and the project boundary. Therefore, all location information (i.e. project area, disturbed acreage, latitude, and longitude) is to be provided by the user. Other required data includes annual rainfall for the project site along with land cover, soil, and watershed information. Many of these data are provided in federal databases exposed using web services. The following section describes these web services and how they can be used for data access to support this stormwater management application.

Table 1. Information needed to support a nutrient trading DSS.

Required Information	Data Needed	Source	Web Service
Project Area		User input	n/a
Disturbed Area		User input	n/a
Location (City/County)		User input	n/a
Latitude and Longitude		User input	n/a
4th Order HUC		NHD	WATERS
Nutrient TMDL		303(d) List	WATERS
VRRM Data Inputs	Annual rainfall	Watershed characteristics	StreamStats
	Land Cover	NLCD	No
	Soils Information	SSURGO	Soils Data Access
TR-20 Data Inputs	Drainage Area	Watershed boundary	StreamStats
	Channel Length	NHD	WATERS
	Rainfall Amount	Watershed characteristics	StreamStats
	Soils Information	SSURGO	Soils Data Access
	Land Cover	NLCD	No
Pricing		User input	n/a
Bank Locations		User input	n/a

Service Descriptions

Three service providers are available to provide much of the data required to build a DSS for the example application (Table 1): EPA WATERS, USDA Soils Data Access, and USGS StreamStats. Each of the services are maintained by a federal agency and are open for public access.

EPA WATERS

EPA's Watershed Assessment, Tracking and Environmental Results System (WATERS) provides water quality information from various EPA sponsored programs and links it to the national surface water network (Environmental Protection Agency, 2015). The surface water network is based on the NHDPlus dataset. Users can locate dischargers, view water quality monitoring results, impaired water reports, and perform general stream navigation. The web services made available by WATERS also expose components used to perform complex analyses on the supporting datasets, such as National Hydrography Dataset (NHD), NHDPlus, and the Watershed Boundary Dataset (WBD). Included in these services are the point indexing service and the upstream/downstream search service, among several others. The point indexing service links a coordinate location (expressed as a latitude and longitude) to the NHDPlus flow line network. The upstream/downstream search service provides the ability to navigate upstream or downstream a user provided distance from a point on the network and returns a list with any events encountered during the traversal.

USGS StreamStats

StreamStats was developed by the USGS to provide users with several analytical tools that are useful for water resource planning, engineering, and design purposes (United States Geological Survey, 2016). The web services provided through StreamStats can be accessed using the StreamStats service browser interface or simply through a web browser. The URL for a service request includes the service name, inputs required by the service, optional response formats, and what parameters the client wants to include in the output. These services allow the client to obtain the basin boundary, characteristics, and streamflow for a selected location on a stream network.

StreamStats is built partly on ArcHydro, a data model and tools for hydrologic data processing within a Geographic Information System (GIS) (Maidment, 2002). Access through the network is provided though ArcServer. Elevation data is derived from the National Elevation Dataset (NED) and adjusted so that the stream channels correspond to those represented in the high-resolution version of the NHD, and so that watersheds correspond to those delineated in the WBD. After basin characteristics are measured, values are input to the National Streamflow Statistics Program, which is a program that uses USGS regression equations to estimate streamflow statistics for points along the river network.

USDA Soils Data Access

USDA's Soil Data Access web services were developed in order to meet objectives that were not being met by the Web Soil Survey and the Geospatial Data Gateway (United States Department of Agriculture, 2016). One of the objectives was to provide a way to request the data, ad-hoc, for any size area of interest and in real-time. Currently, the Soil Data Access services return spatial and tabular data using separate requests. The user must first define the area of interest and request the spatial data then use the map unit keys from within the spatial data to request the desired tabular data.

Soils Data Access offers several options for accessing the spatial and tabular data. Users can access tabular data via SOAP or REST/POST requests. Instead of tabular data, users can request spatial data using different coordinate systems including WGS84, NAD83, UTM, and Web Mercator. The tabular service includes a RunQuery method that returns XML data for one or more SQL statements. The spatial services follow the Open Geospatial Consortium (OGC) Web Feature Service (WFS) standard and include GetCapabilities, DescribeFeatureType, and GetFeature. For this study, the GetFeature method using the WGS84 coordinate system was used. Two layers accessible through the GetFeature method are the mapunitpoly and mapunitpolyextended layers. The mapunitpoly layer contains identifying information about soil map units and the mapunitpolyextended layer contains more specific information like Hydrologic Soil Group, moisture, and slope.

METHODS

Approach

All testing was done on virtual machines (VMs) provided through Amazon Web Service's (AWS) Elastic Compute Cloud (EC2), specifically the t2.micro instances and the Ubuntu Amazon Machine Image. AWS allows VMs to be created in one of three locations in the US: Washington D.C., Oregon, or Northern California. Unless specified, all experiments were conducted using VMs located in Washington D.C. Once the VM was running, Python scripts for each experiment were moved to the VM's local directory and run from the command line. The *urllib2* library was used to request URLs within the Python scripts. Timers, from the *time* library, were set before and after the URL request was made and returned. The service response time was defined as the difference between when the URL request was made and when the response was returned. Results were output to a comma-separated file that was saved on the VM. Each experiment was run multiple times to access the variability in service requests. After copying the result files to a local computer, Python's *Pandas* library was used to analyze the data and the *matplotlib* library was used to visualize the data. Bar charts with error bars were made for each experiment to show the mean and one standard deviation around the mean for the response times.

One-Way Analysis of Variance (ANOVA) was used to examine whether mean response times between different tests were statistically significant. ANOVA was used to test the null hypothesis that population means for the response time between several groups were equivalent. If this null hypothesis is accepted, then all groups are considered statistically similar. However, if the null hypothesis is rejected, then at least one of the groups is significantly different than the others. ANOVA tests do not indicate which group is different; therefore a post hoc test was required. The Tukey Honest Significant Differences (HSD) test was used to identify groups whose differences exceed the expected standard error, indicating which group is significantly different. Both of these statistical analyses are available through the R software package, which is widely used for statistical computing and graphics (R Development Core Team, 2008). The ANOVA tests and the post hoc testing using the Tukey HSD method were completed using R, version 3.3.1, and an alpha level of 0.05.

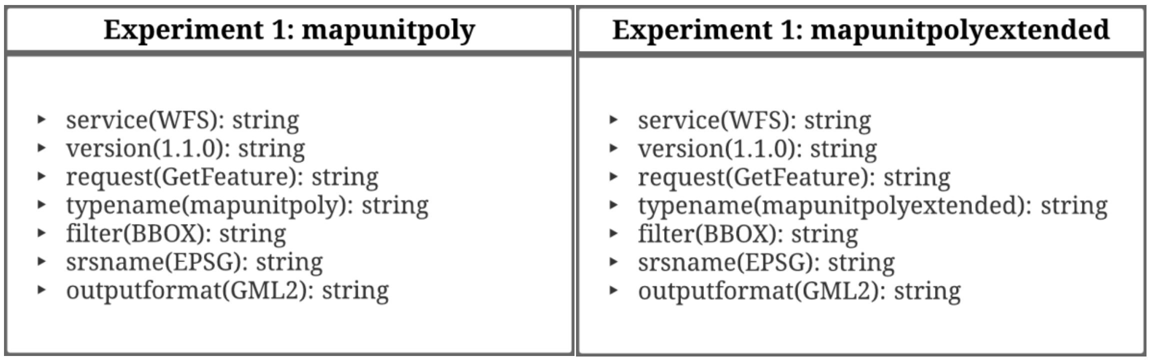
Experiments

Five experiments were designed to test response times for essential data access queries required in the stormwater management application described earlier. Key parameters for each query were varied to measure their impact on response time. Experiment 1 was designed to test how the USDA soils web services responded to increasing study area sizes. Experiment 2 was designed to test how the distance between the project location and the river affected response times for the point indexing service from WATERS. Experiment 3 was designed to test how the downstream search distance affected response times for WATERS' upstream/downstream search service. Experiment

4 was designed to test how the size of the watershed affected response times for StreamStats services. Finally, experiment 5 was designed to test how the client's location affected response times for all of the services. Details of the steps taken within each experiment are described in the following subsections.

Experiment 1: Soils Data Access

In order to conduct experiment 1, two URL requests were made: one for the mapunitpoly service and one for the mapunitpolyextended service. The parameters specified in both requests are summarized in Figure 1. Both URLs request the GetFeature method, output in GML2 format, projection into WGS84 coordinates, and the default service type and version for the Soils Data Access Service (Figure 1). The GetFeature method returns a feature collection for a layer for an area of interest. A bounding box defined the area of interest in this experiment. The coordinates for the bounding box were varied to create polygons of 1, 10, 100, 1,000, and 10,000 acres to test the response times as the study area increases. As shown in Figure 2, the mapunitpoly service was requested first followed by the mapunitpolyextended service after the mapunitpoly response was returned. Each URL was called 100 times in this experiment to measure the variability in response times.



Experiment 1: mapunitpoly	Experiment 1: mapunitpolyextended
▸ service(WFS): string	▸ service(WFS): string
▸ version(1.1.0): string	▸ version(1.1.0): string
▸ request(GetFeature): string	▸ request(GetFeature): string
▸ typename(mapunitpoly): string	▸ typename(mapunitpolyextended): string
▸ filter(BBOX): string	▸ filter(BBOX): string
▸ srsname(EPSG): string	▸ srsname(EPSG): string
▸ outputformat(GML2): string	▸ outputformat(GML2): string

Figure 1. URL parameters for the web services tested in experiment 1.

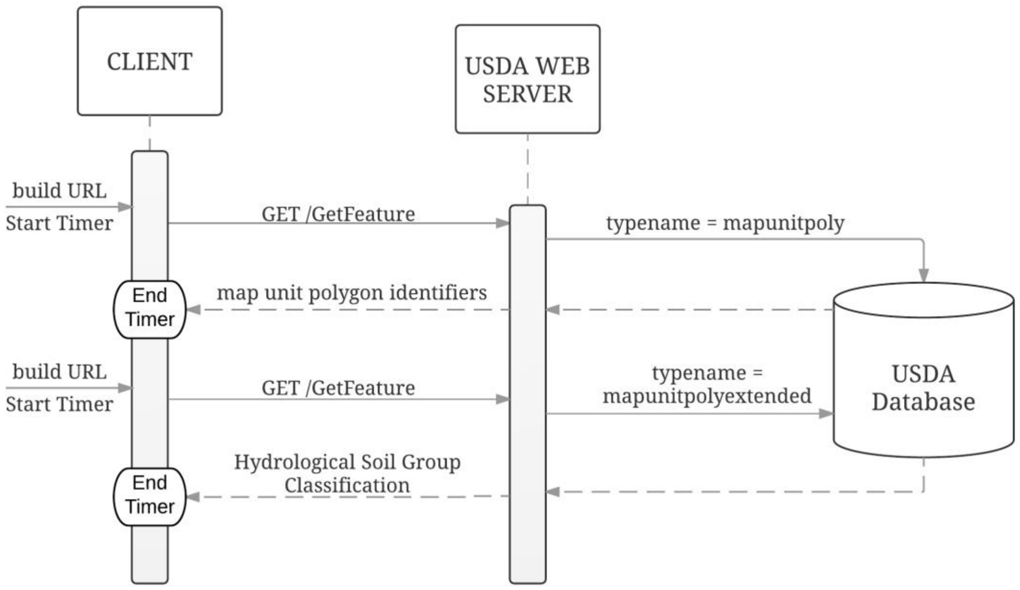
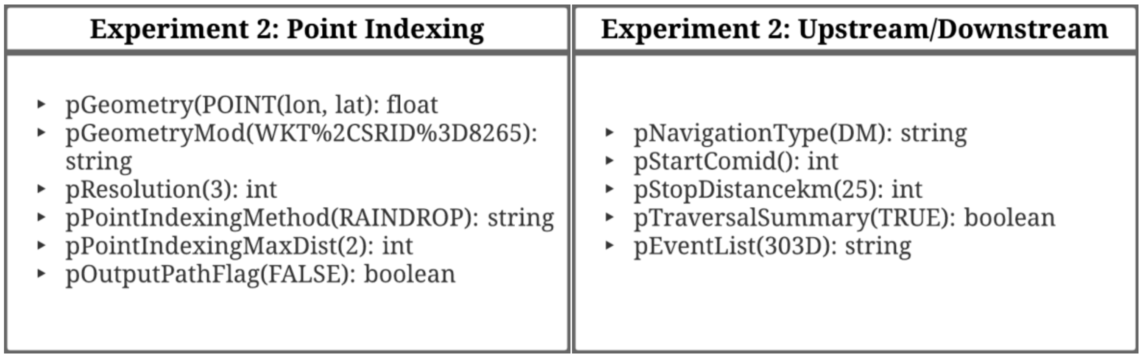


Figure 2. Sequence diagram for web services tested in experiment 1.

Experiment 2: Distance from Stream Network

Two URL requests were made in experiment 2: one for the point indexing service and one for the upstream/downstream search service. The point indexing method was set to RAINDROP mode in order to force downhill travel to the river network. The point geometry represents the starting location that could be a project’s most downhill point. The maximum indexing distance was set to the default value of 2 km as shown in Figure 3. The OutputPathFlag was set to FALSE. The coordinates for the point geometry represent distances 0.5, 1, 2, 3, and 4 km away from the nearest, downhill flowpath. The upstream/downstream search service was given a start COMID value, a unique identifier for a feature within the National Hydrography Dataset (NHD), which was obtained from the output of the point indexing service.



Experiment 2: Point Indexing	Experiment 2: Upstream/Downstream
‣ pGeometry(POINT(lon, lat): float ‣ pGeometryMod(WKT%2CSRID%3D8265): string ‣ pResolution(3): int ‣ pPointIndexingMethod(RAINDROP): string ‣ pPointIndexingMaxDist(2): int ‣ pOutputPathFlag(FALSE): boolean	‣ pNavigationType(DM): string ‣ pStartComid(): int ‣ pStopDistancekm(25): int ‣ pTraversalSummary(TRUE): boolean ‣ pEventList(303D): string

Figure 3. URL parameters for the web services tested in experiment 2.

The stop distance was set to a constant 25 km from the initial point indexing location for all trials. The traversal summary was set to be downstream, mainstem (DM). The traversal summary, flowline summary, and 303d event summary lists were returned for all trials. The URL for point indexing was built and requested first. Upon return of the data for the point indexing service, the URL for the upstream/downstream search service was built and requested (Figure 4). Both URLs were requested 100 times for this experiment to measure the variability in response time.

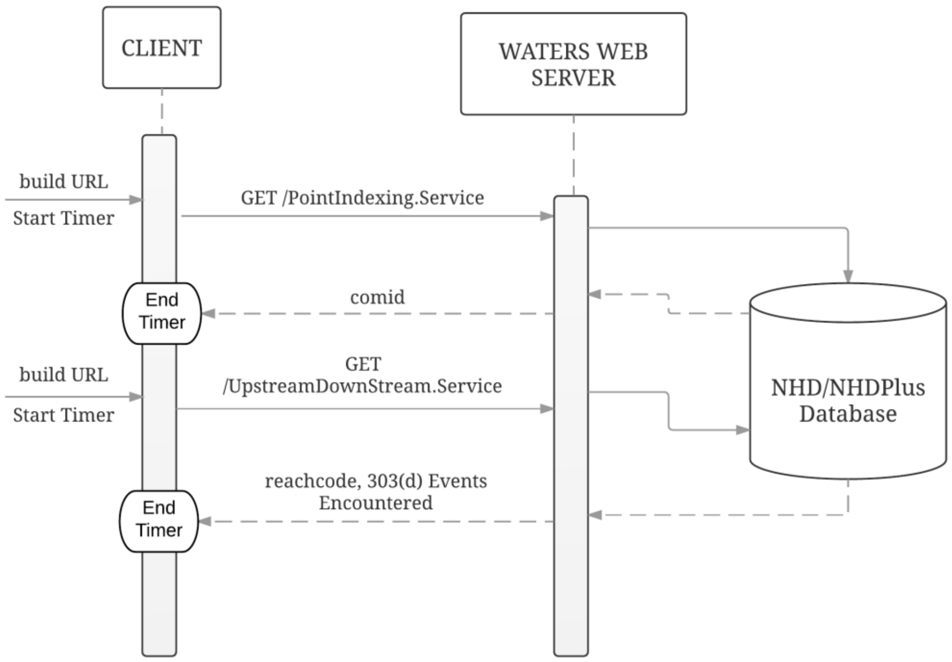
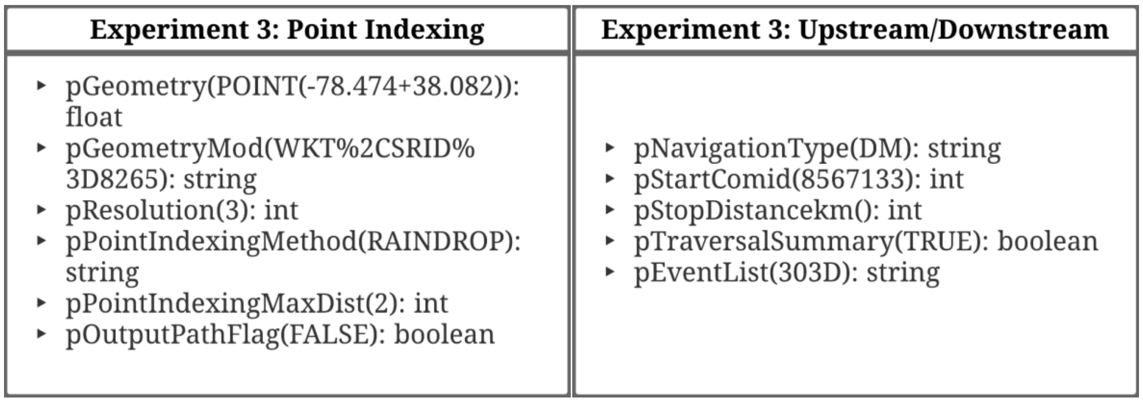


Figure 4. Sequence diagram for the web services tested in experiments 2 and 3.

Experiment 3: Stream Network Search Distance

Two URL requests were made in this experiment: one for the point indexing service and one for the upstream/downstream search service. The URL parameters are summarized in Figure 5. The downstream search distance was varied between 1, 5, 10, 25, and 50 km for the trials. All other variables used in experiment 2 were held constant in experiment 3. In contrast to experiment 2, the input point geometry was set to a single location that was constant for all trials. The service call and timing sequence remained the same as experiment 2 and are detailed in Figure 4. Each URL was requested 50 times for this experiment.



Experiment 3: Point Indexing

- pGeometry(POINT(-78.474+38.082)): float
- pGeometryMod(WKT%2CSRID%3D8265): string
- pResolution(3): int
- pPointIndexingMethod(RAINDROP): string
- pPointIndexingMaxDist(2): int
- pOutputPathFlag(FALSE): boolean

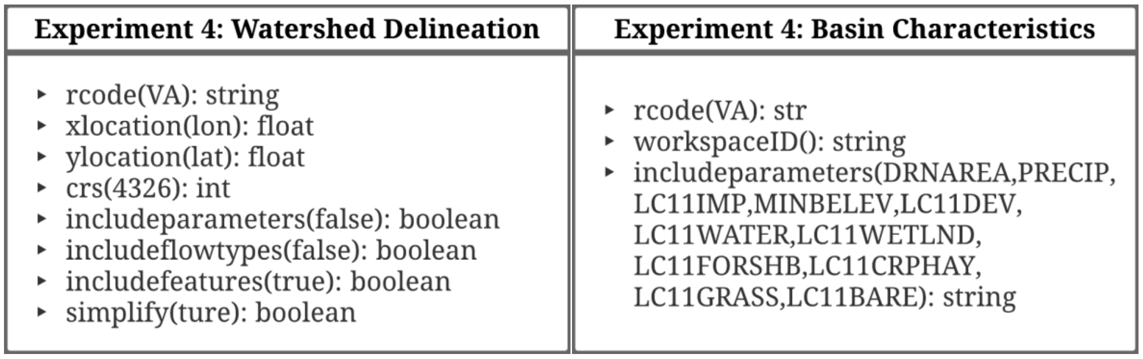
Experiment 3: Upstream/Downstream

- pNavigationType(DM): string
- pStartComid(8567133): int
- pStopDistancekm(): int
- pTraversalSummary(TRUE): boolean
- pEventList(303D): string

Figure 5. URL parameters for the web services tested in experiment 3.

Experiment 4: Watershed Properties

Two URL requests were made in experiment 4: one for watershed delineation and the other for basin characteristics of the watershed. The URL parameters are summarized in Figure 6. The coordinates for x-location and y-location represent the latitude and longitude of a point on the stream network. For this experiment, the coordinates were varied to produce watersheds of 200, 800, 2,500, 25,000, and 110,000 acres. Both services offer options to include different lists in the output. The URLs shown in Figure 6 detail the selected lists. Upon completion, the watershed delineation service returned a workspaceID. The workspaceID was used in the basin characteristics service to return specific watershed information. For this experiment, only a select number of basin characteristics were returned including: drainage area, annual average precipitation, minimum elevation, and National Land Cover Database 2011 land cover percentages. These characteristics were chosen using the includeparameters setting. The URL for the watershed delineation service was written, requested, and returned before the same process was initiated for the basin characteristics service so that once one service request was in process at any given time (Figure 7). The URLs were only called 25 times due to longer response times in this experiment.



Experiment 4: Watershed Delineation

- rcode(VA): string
- xlocation(lon): float
- ylocation(lat): float
- crs(4326): int
- includeparameters(false): boolean
- includeflowtypes(false): boolean
- includefeatures(true): boolean
- simplify(ture): boolean

Experiment 4: Basin Characteristics

- rcode(VA): str
- workspaceID(): string
- includeparameters(DRNAREA,PRECIP, LC11IMP,MINBELEV,LC11DEV, LC11WATER,LC11WETLND, LC11FORSHB,LC11CRPHAY, LC11GRASS,LC11BARE): string

Figure 6. URL parameters used for the web services tested in experiment 4.

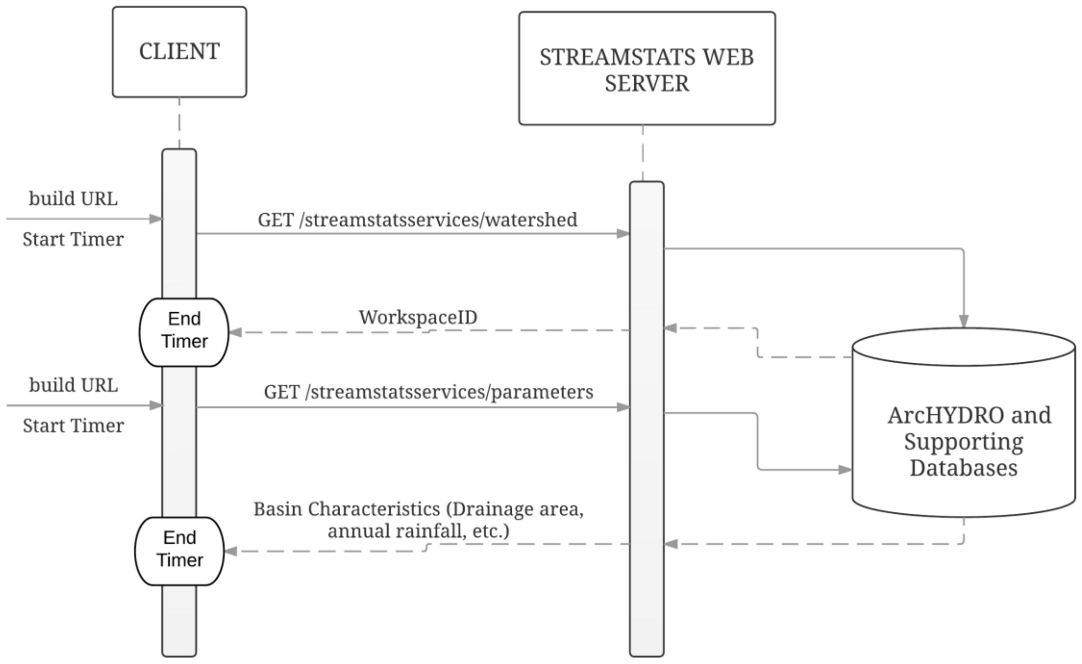

CLIENT
STREAMSTATS WEB SERVER
build URL
Start Timer
GET /streamstatsservices/watershed
End Timer
WorkspaceID
build URL
Start Timer
GET /streamstatsservices/parameters
ArcHYDRO and Supporting Databases
Basin Characteristics (Drainage area, annual rainfall, etc.)
End Timer

Figure 7. Sequence diagram for the web services tested in experiment 4.

Experiment 5: Location of Server and Client Machines

Virtual Machines (VMs) in three different geographic locations, Washington D.C., Oregon, and Northern California, were used in this experiment as the client machine for making the service requests. Scripts for the USDA, EPA WATERS, and

StreamStats were started at the same time in the three different locations. These scripts were the same form as the first four experiments. The URL parameters were set to general values and kept constant, so that only the client location was varied for this experiment. The Soils Data Access service was bounded by coordinates representing an approximately 25,000 acre polygon. All other parameters remained the same in experiment 1. EPA WATERS was given the same starting location as experiment 3. The downstream search distance was set to 25 km, the navigation type to downstream, mainstem, and the 303(d) event list was populated for all trials. Finally, the StreamStats watershed was the same one as the 2,500 acre watershed tested in experiment 4. The sequence for this experiment follows that of the first four experiments. The Soils Data Access and WATERS URLs were requested 100 times while the StreamStats URLs were only requested 25 times due to time constraints. Three VM's were initiated, one in each geographic location, and Python scripts for the individual service to be tested were loaded onto the VM's local directory. In order to minimize the effect of network traffic differences at the three locations, all tests in experiment 5 were run at approximately 1 p.m. Eastern time during a workday. Results were stored separately and copied into a single file after the scripts had completed.

RESULTS

The results of experiment 1 show how the requested polygon size affects the response times for the Soils Data Access web services (Figure 8). For polygons 1,000 acres and smaller, the requested information was returned in under 1 second for both of the layers requested. The response time is approximately three times slower for the mapunitpoly service and ten times slower for the mapunitpolyextended service when requesting data for a 10,000 acre polygon compared to the 1, 10, and 100 acre polygons.

A one-way ANOVA showed that the polygon size did have a significant effort on response times for the mapunitpoly layer when comparing all four polygon sizes at the $p < .05$ level [$F(4,495) = 2779$, $p \sim 0$]. Post hoc comparisons indicated that the 1, 1,000, and 10,000 acre polygon sizes were significantly different from the others, but that the 10 and 100 acre polygon response times did not significantly differ. There was also a significant effect of polygon size on response times for the mapunitpolyextended layer at the $p < 0.05$ level [$F(4,495) = 4678$, $p \sim 0$]. Post hoc comparisons indicated that the 1, 10, and 100 acre polygon response times were not significantly different from one another, while the 1,000 and 10,000 acre polygons were significantly different.

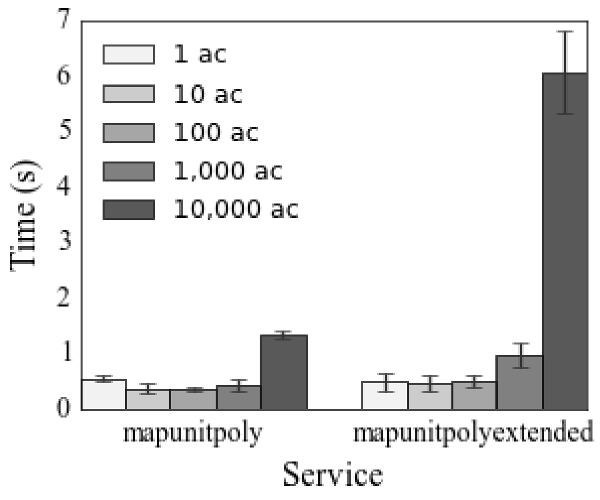


Figure 8. Average response times and standard deviations experiment 1.

The results of experiment 2 show how the requested starting location, specifically the distance away from the stream network, affects response times for the EPA

WATERS' web services (Figure 9). There was approximately a 25% increase in response times for points 3 - 4 km away from the stream network when compared to the points 0.5 - 2 km from the stream network. The point indexing service took 1 – 2 seconds longer than the upstream/downstream search service. A one-way ANOVA used to compare the effect of initial distance from a flowline on response times for WATERS services showed there was no significant effect of starting coordinates on the response times for the upstream/downstream search service at the $p < 0.05$ level [$F(4,495) = .244$, $p < 0.913$]. The upstream/downstream search service returned times just under 1.5 seconds. There was a significant effect of starting location on response times for the point indexing service at the $p < 0.05$ level [$F(4,495) = 81.12$, $p \sim 0$]. Post hoc tests indicated that the ½, 1, and 2 km distances were not significantly different from one another, and that the 3 and 4 km distances were not significantly different from one another.

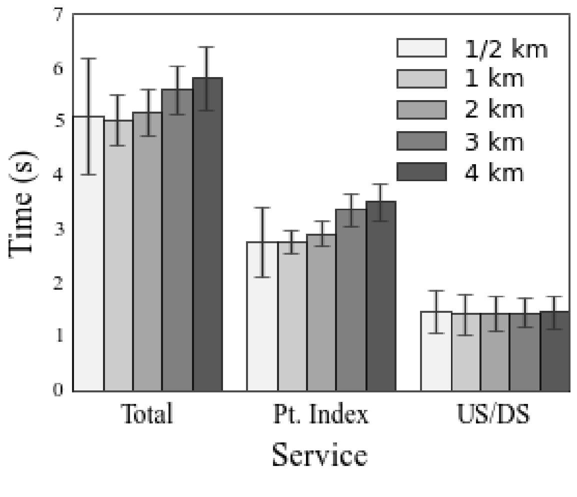


Figure 9. Average response times and standard deviations experiment 2.

The results of experiment 3 show how the requested downstream search length affects response times for the EPA WATERS web services (Figure 10). The 50 km search distances took approximately seven times longer to return compared to the 1 km search distances. The ANOVA indicated that the response times for the point indexing service were not significantly different at a p < 0.05 level [F(4,245) = .423, p < 0.792]. However, there was a significant effect of search distance on response for the upstream/downstream search service at a p < 0.05 level [F(4,245) = 153.7, p ~ 0]. Post hoc tests indicated that the 1 and 5 km response times were not significantly different from each other and that the 25 and 50 km response times were not significantly different from each other.

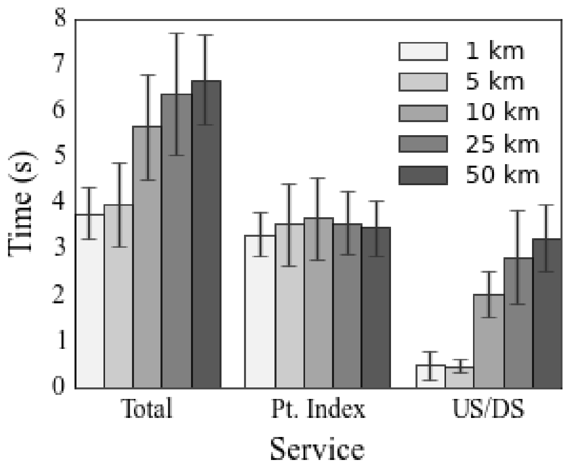


Figure 10. Average response times and standard deviations experiment 3.

The upstream/downstream search service starts at coordinates on a flowline and then travels a specified distance downstream and outputs events that are encountered, in this case any 303(d) listings. To determine if response times were due to more events being encountered and therefore the message size increasing, the response message size was also recorded (Table 2). Internet speeds were on the order of 10 – 100 Mbits/second for the t2.micro machine. The approximately 54 kB increase in message size between the

1 km and 50 km search distance would only justify an increase of 0.005 to 0.05 seconds in response time, thus the increasing message size was not a significant cause for the increased response time.

Table 2. Returned file sizes for Experiment 3.

Service	Search Distance	File Size (kB)
Point Indexing	1	1.5
	5	1.5
	10	1.5
	25	1.5
	50	1.5
Upstream/Downstream	1	5.3
	5	8.6
	10	11.6
	25	34.1
	50	59.2

The results of experiment 4 show how the size of the delineated watershed affects response times for the StreamStats web services (Figure 11). Response times were relatively constant despite the fact that the watershed size increased substantially. This was validated with the ANOVA test at a $p < 0.05$ level [F(4,115) = .493, $p < 0.741$] for the watershed delineation service as well as for the basin characteristics service [F(4,115) = 1.784, $p < .137$]. The watershed delineation responded around 40 seconds while the basin characteristics were returned within 15 – 20 seconds for all watershed sizes tested. Despite scaling well to increasing study area sizes, these services were the slowest of the ones tested by a factor of nearly ten.

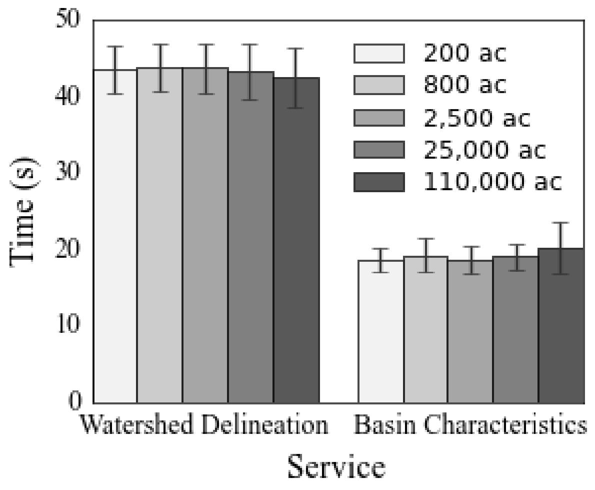


Figure 11. Average response times and standard deviations experiment 4.

The results of experiment 5 show how the client's location affects response times for all service providers tested (Figure 12). ANOVA tests indicated that all services tested were significantly affected by the client location at the $p < 0.05$ level. The Washington D.C. location had longer response times for the Soils Data Access services and for the StreamStats services when compared to the two western U.S. locations. However, it had shorter response times compared to the two western locations for the WATERS services. Post hoc testing indicated that the two locations in the western United States were not significantly different for most services, except for the soils services, with the Northern California location returning responses more quickly than the Oregon location.

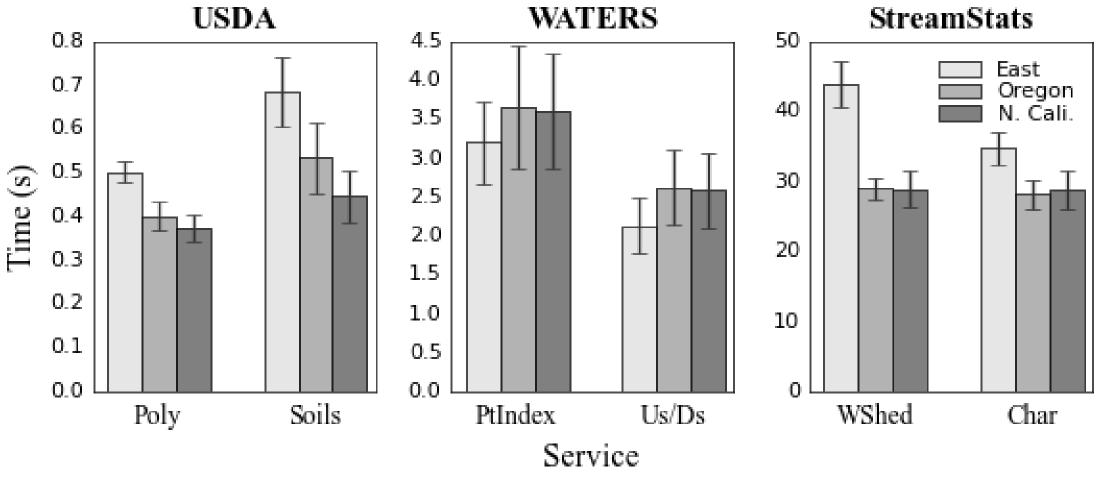


Figure 12. Average response times and standard deviations experiment 5.

DISCUSSION

Performance of Services

To provide some context for the response times for each of these services, Nielsen (1993) presents three thresholds for web application response times. At and below 0.1 seconds, the user feels that the application is reacting instantaneously. One second represents the limit to keep user's attention uninterrupted, although the delay is noticeable. Ten seconds is the limit for keeping a user's attention on the application. Returns longer than ten seconds should have some type of progress icon or offer asynchronous capabilities. In a more recent update, Nielsen (2014) stands by his usability recommendations as they are based on user experience and not the performance of an application. These thresholds are meant to be guidelines to use when optimizing application performance. For the purposes of this discussion, 10 seconds was selected as the upper limit for a distributed DSS with on-demand access to data from web services. It is important to keep in mind that these services are automating complex and tedious tasks that, when traditionally performed with desktop computers, and can take several minutes and even hours to complete depending on the size of the data and number of processing steps required. Thus, while 10 seconds may seem very quick for performing a task like watershed delineation, in a distributed, on-demand DSS, having a response within 10-seconds would allow for a more interactive user experience compared to waiting longer for the response.

The two Soils Data Access web services had the shortest response times. In most cases, except for the 10,000 acre study area, both services returned before the one second threshold. The mapunitpoly service consistently returned faster than the mapunitpolyextended service, with the difference increasing as the size of the polygon increased. The two WATERS services tested were the next fastest services to return. The point indexing service was slower than the upstream/downstream search service, however, as the downstream search distance increases, the two services' response times approached each other. The two StreamStats services were the slowest to return of the ones tested with both being over the ten second threshold for the watershed sizes tested.

To reiterate, the experiments were designed to attempt to identify parts of the services where potential bottlenecks may occur, along with general response times for the services. The most obvious impediment to an on-demand DSS in the services tested would be the response times for the StreamStats web services. However, these services are also performing arguably the most significant data processing before returning information to the user. Innovative methods to speed up these services, however, would benefit on-demand applications. Another area for improvement would be for soil data requests for large areas. The Soils Data Access mapunitpolyextended service took six times longer to obtain soils characteristics for 10,000 acre study areas when compared to 1,000 acre study areas. For the application that motivated this study, projects would most likely be less than 1,000 acres, however, other cases may require soil data for larger areas. Although the initial distance away from the stream network does affect response times, the effect is small relative to other response times. However, the search distance downstream should be considered within a DSS utilizing these services. The response times were almost seven times longer for a 50 km search distance when compared to the 1 km search distance. Therefore, it may be necessary to maintain a maximum downstream search distance that users cannot exceed.

For all six web services, but especially the two provided by StreamStats, there was a significant difference between response times from client VM's located in different parts of the country. This result suggests that efforts to route service requests to more nearby servers could have a significant impact on response times. Other factors such as network traffic may also be playing a role in these results, however, and further testing could better pinpoint the exact cause for these delayed response times. However, what can be taken away from the experiment is that the client's location affects web service response times in a somewhat surprisingly significant way. For example, services offered by StreamStats experienced a 50% reduction in response times when the client was located in the western United States compared to the eastern United States (Figure 12).

Variability, Reliability and Limitations of Services

StreamStats services were the most variable, followed by WATERS and Soils Data Access services respectively. StreamStats services' deviations exceeded 1 second in all cases (Figures 11 & 12). WATERS services' deviations ranged between 0.2 and 1 second (Figures 9, 10 & 12). Soils Data Access mapunitpoly service was consistently under 0.1 second while mapunitpolyextended was under 0.2 second for most trials (Figures 8 & 12). Generally, the standard deviations were within 5-25% of the mean response times.

Reliability concerns that affect DSS implementation include timeouts and no data or wrong data returns. In addition to the requests made for the actual experiments, these services were also requested many times during set-up and code debugging. Hundreds of requests were made to each of the services and no timeouts or data errors were encountered. StreamStats is under active development with the USGS rolling out updates for each state and beta versions being tested, with one just being made available publically. In the past, WATERS has retired services that were considered obsolete or unpopular. This could become a problem if distributed DSSs depend on services for reproducing past studies and better ways to maintain or archive legacy services is an important area for future work. The Soils Data Access is taken offline for maintenance from 12am to 4am Mountain time every night in order to complete updates. Overall, although no issues were specifically encountered during testing, there are conditions that need to be considered before using these web services within production systems.

Remaining Barriers to Achieving the On-Demand Environmental DSS

In regards to an environmental DSS to support the stormwater management application, there are several areas that need attention. The first area is required data that is unavailable or not easily obtained through web services. I was unable to find an authoritative web service for land cover data. This kind of data is similar to that of the soils data in that it can take both a spatial and a tabular form. StreamStats does provide some land cover derived data within its basin characteristics service. However, this is not consistent for all states. Other data needed for the scenario were more localized data such as the locations of nutrient credit banks, amount of available credits, and pricing. Although federal data providers are making increased use of web services and their adoption at local governmental levels is growing for standardized data like geospatial data layers, more specialized datasets without well-established standards are still a work in progress. For example, nutrient credit prices in Virginia are determined for each major river basin and managers have to contact the bank to discuss how many credits are available at what cost. This type of information would be ideal for access through a web service. Prices may change overnight but as long as the information was updated by the service provider, the consumer (in this case a project manager interested in purchasing credits) could make a request using web services and be provided with current and accurate information.

The second area for improvement is to the service response times. The USDA web services are suitable for use within a distributed DSS because their response times were consistently below 1 second for most tests. It is only when the study area becomes very large (10,000 acres in our analysis) that the service slows to a point where on-demand access is no longer practical. The WATERS services returned responses within 5 – 6 seconds typically, except in the cases where the downstream search length was below 5 km or above 10 km. These response times could also be acceptable for on-demand access, especially if there is a progress bar for users. StreamStats response times were too long for an on-demand application (often over 40 seconds for the watershed delineation). Although StreamStats services are performing complex calculations that would take significant time if performed manually, they are not yet at the point where on-demand applications can make use of the services without an effort to make sure users stay engaged during the waiting time (Nielsen, 1994).

In order to create an environmental DSS using on-demand web services, the response times for the services should improve and more attention should be paid to methods for speeding up complicated analysis services like those provided through StreamStats or Soils Data Access for large areas. Improving the response times may be as simple as using larger machines to handle requests, improving database performance, or other methods for optimizing the existing software. Experiment 5 demonstrated that the client location had an effect on response times: consistently a 10-20% difference in response times and in some cases as much as a 50% difference. This suggests data transmission over the Internet may also a bottleneck in these applications. The message sizes passed between clients and servers could potentially be more streamlined in order to reduce these network transmission times. Or servers could be geographically distributed and client requests routed to geographically nearby servers if message sizes cannot be further reduced.

CONCLUSIONS

This research was conducted to explore the feasibility of using available web services from federal agencies to support a distributed environmental DSS with on-demand data access. The popularity of web applications has given rise to web services as a new area for information dissemination due to the benefits offered in interoperability, access, and standardization. Previous work has been done on using web services for historical data access, analysis, visualization, and simulation of environmental processes. Most of the work has been focused on the design or implementation of the services themselves. There has been less work on examining the services' performance for building a distributed environmental DSS application. A stormwater management application was used to define typical service requests for an environmental DSS. Experiments were designed to test potential bottlenecks for service requests from the three major federal agencies.

The Soils Data Access services averaged response times below 1 second for study areas 1,000 acres and below. The WATERS services responded in between 4 – 7 seconds. The initial distance away from the stream network did not impact response times as much as the downstream search distance. The StreamStats services responded in 40 seconds on average for basin delineations and in between 15 – 30 seconds for the basin characteristics, depending on the number of parameters requested. The StreamStats service response times were relatively constant for increasing watershed sizes. Client location did have an effect on the response times for all three services with generally a difference of 10-20% and as much as 50% in response times.

The variability in response time for the same service call repeated at different times ranged from less than a tenth of a second to a few seconds, depending on the service. Generally, one standard deviation of the service calls was 5-25% of the mean response time for each service. Reliability concerns stemming from timeouts or requests not being returned were not encountered. The USDA web services are already suitable for a distributed environmental DSS due to their short response times for typical project study areas. The WATERS services may also be suitable, however additional improvement in response times and variability would be beneficial in order to reduce response times to below 1 second. The StreamStats services are less suitable for a distributed DSS application due to long response times of over 40 seconds. When using the StreamStats service, a progress bar would be necessary to inform users of wait times.

Although the services tested are making progress toward the long term vision of distributed environmental DSSs, there are opportunities for future research and development. Future work should be devoted to creating new web services needed for environmental DSS applications. For example, services for land use and land cover data and more localized information. There is also a need for improvement to the algorithms supporting more rapid watershed delineation, especially for small watersheds. Web services providers may also benefit from geographically distributing their services in order to handle requests from different parts of the country more quickly, since this was also found to be a significant factor in service response time.

REFERENCES

Ames, D.P., Horsburgh, J.S., Cao, Y., Kadlec, J., Whiteaker, T., Valentine, D., 2012. HydroDesktop: Web services-based software for hydrologic data discovery, download, visualization, and analysis. Environ. Model. Softw. 37, 146–156. doi:10.1016/j.envsoft.2012.03.013

Barry, D.K., (2016). Service architecture. Retrieved from http://www.service-architecture.com/articles/web-services/index.html

Booth, N.L., Everman, E.J., Kuo, I.L., Sprague, L., Murphy, L., 2011. A Web-Based Decision Support System for Assessing Regional Water-Quality Conditions and Management Actions. J. Am. Water Resour. Assoc. 47, 1136–1150. doi:10.1111/j.1752-1688.2011.00573.x

Buytaert, W., Baez, S., Bustamante, M., Dewulf, A., 2012. Web-based environmental simulation: Bridging the gap between scientific modeling and decision-making. Environ. Sci. Technol. 46, 1971–1976. doi:10.1021/es2031278

Díaz, L., Granell, C., Gould, M., 2008. Case Study: Geospatial Processing Services for Web-based Hydrological Applications. Geospatial Serv. Appl. Internet 10, 31–47. doi:10.1007/978-0-387-74674-6_2

Environmental Protection Agency. (2015). WATERS web services. Accessed on June 1, 2016 from https://www.epa.gov/waterdata/waters-web-services

Feng, M., Liu, S., Euliss, N.H., Young, C., Mushet, D.M., 2011. Prototyping an online wetland ecosystem services model using open model sharing standards. Environ. Model. Softw. 26, 458–468. doi:10.1016/j.envsoft.2010.10.008

Fielding, R.T. (2000). Architectural styles and design of network-based software architectures (Doctoral dissertation, University of California, Irvine).

Goodall, J.L., Horsburgh, J.S., Whiteaker, T.L., Maidment, D.R., Zaslavsky, I., 2008. A first approach to web services for the National Water Information System. Environ. Model. Softw. 23, 404–411. doi:10.1016/j.envsoft.2007.01.005

Goodall, J.L., Robinson, B.F., Castronova, A.M., 2011. Modeling water resource systems using a service-oriented computing paradigm. Environ. Model. Softw. 26, 573–582. doi:http://dx.doi.org/10.1016/j.envsoft.2010.11.013

Granell, C., Diaz, L., Gould, M., 2010. Service-oriented applications for environmental models: Reusable geospatial services. Environ. Model. Softw. 25, 182–198. doi:10.1016/j.envsoft.2009.08.005

Hey, T., Trefethen, A.E., 2005. Cyberinfrastructure for e-Science. Science (80-.). 308, 817–821.

Horsburgh, J.S., Tarboton, D.G., Piasecki, M., Maidment, D.R., Zaslavsky, I., Valentine, D., Whitenack, T., 2009. An integrated system for publishing environmental observations data. Environ. Model. Softw. 24, 879–888. doi:10.1016/j.envsoft.2009.01.002

Horsburgh, J.S., Tarboton, D.G., Schreuders, K.A.T., Maidment, D.R., Zaslavsky, I., Valentine, D., 2010. HydroServer: A Platform for Publishing Space-Time Hydrologic Datasets, in: AWRA 2010 Spring Specialty Conference - GIS and Water Resources.

Laniak, G.F., Olchin, G., Goodall, J., Voinov, A., Hill, M., Glynn, P., Whelan, G., Geller, G., Quinn, N., Blind, M. and Peckham, S., 2013. Integrated environmental modeling: a vision and roadmap for the future. Environ. Model. Softw. 39, 3-23.

Maidment, D.R., 2002. *Arc Hydro: GIS for water resources* (Vol. 1). ESRI, Inc..

Matthies, M., Giupponi, C., Ostendorf, B., 2007. Preface - Environmental decision support systems: Current issues, methods and tools. Environ. Model. Softw. 22, 123–127. doi:10.1016/j.envsoft.2005.09.005

Microsoft Developer Network. (n.d.). Chapter 1: service-oriented architecture (SOA). Accessed on June 1, 2016 from https://msdn.microsoft.com/en-us/library/bb833022.aspx

Mineter, M.J., Jarvis, C.H., Dowers, S., 2003. From stand-alone programs towards grid-aware services and components: A case study in agricultural modelling with interpolated climate data. Environ. Model. Softw. 18, 379–391. doi:10.1016/S1364-8152(02)00073-7

Nielson, J. (1993). Usability engineering. San Franciso, CA: Morgan Kaufmann Publishers Inc.

Nielsen, J. (2014). Response times: the 3 important limits. Nielsen Norman Group. Accessed from https://www.nngroup.com/articles/response-times-3-important-limits/

The Open Group. (2016). Service oriented architecture: what is SOA? Retrieved from http://www.opengroup.org/soa/source-book/soa/soa.htm

Philip Chen, C.L., Zhang, C.Y., 2014. Data-intensive applications, challenges, techniques and technologies: A survey on Big Data. Inf. Sci. (Ny). 275, 314–347.

doi:10.1016/j.ins.2014.01.015

R Development Core Team, (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, Accessed from http://www.R-project.org

Sun, A., 2013. Enabling collaborative decision-making in watershed management using cloud-computing services. Environ. Model. Softw. 41, 93–97.

Szalay, A., Gray, J., 2006. Science in an exponential world. Nature 440, 413–414.

Tarboton, D.G., Idaszak, R., Horsburgh, J.S., Heard, J., Ames, D., Goodball, J.L., Merwade, V., Couch, A., Arrigo, J., Hooper, R., Valentine, D., Maidment, D.R., 2014. HydroShare: Advancing Collaboration through Hydrologic Data and Model Sharing, in: Ames, D.P., Quinn, N.W.T., Rizzoli, A.E. (Eds.), International Environmental Modelling and Software Society (iEMSs) 7th International Congress on Environmental Modelling and Software. doi:978-88-9035-744-2

United States Department of Agriculture, n.d. Soil Data Access web service help. Accessed on June 1, 2016 from http://sdmdataaccess.nrcs.usda.gov/WebServiceHelp.aspx

United States Geological Survey, n.d. StreamStats service documentation. Accessed on June 1, 2016 from http://streamstatsags.cr.usgs.gov/streamstatsservices/#/

Walker, J.D., Chapra, S.C., 2014. A client-side web application for interactive environmental simulation modeling. Environ. Model. Softw. 55, 49–60.