Design and evaluation of an Open Web Platform cartography lab curriculum

Carl M. Sack and Robert E. Roth

University of Wisconsin Cartography Lab, Department of Geography, University of Wisconsin-Madison, Madison, WI, USA

ABSTRACT
Recent shifts in web map technology away from proprietary software and toward development on the Open Web Platform have increased the number and complexity of technical skills needed to do cartography on the Web. Web-based cartography curricula likewise must be adapted to prepare geography, cartography, and GIS students with the skills needed to make use of open source technology. This paper describes the design and evaluation of a novel curriculum for the laboratory component of a web mapping course offered by a university geography department. We drew from constructivist educational theory to create a scaffolded and spiralled lab curriculum that builds on prior understanding and progressively increases student independence and confidence. We evaluated the success of the new curriculum through an instructor log, student feedback on assignments, and an exit survey. The evaluation revealed significant growth in student abilities and confidence in the use of Open Web Platform-based mapping tools as a result of the curriculum scaffolding. This research provides a methodology for designing and evaluating curriculum around highly technical skills that are increasingly in demand in research, education, and industry careers.

Introduction
Interactive and open source web technologies have profoundly changed the production and distribution of geographic information via maps. Apple Maps alone served more than 5 billion map requests per week in 2015, and thematic map producers in both the public and private sectors – from The New York Times to the U.S. Geological Survey – are putting their maps online as interactive web applications (Jesdanun, 2015). While the growth of open, online, and interactive platforms presents exciting prospects for map design and use, educators in cartography and GIScience have struggled to keep pace with the rapid technological evolution taking place in the digital age (Roth, Donohue, Sack, Wallace, & Buckingham, 2014). In particular, the shift to the Open Web Platform – the standard in-browser development and publishing environment for online maps – increased the range

ARTICLE HISTORY
Received 13 January 2016
Accepted 28 June 2016

KEYWORDS
cartography; web mapping; open source; Open Web Platform; constructivism; scaffolded instruction; spiral curriculum; mixed methods

CONTACT
Carl M. Sack, cmsack@wisc.edu

Supplemental data for this article can be accessed at http://dx.doi.org/10.1080/03098265.2016.1241987.
of technical competencies students need to learn in order to make web maps (Peterson, 2014), forcing educators to rethink the scope and sequence of topics introduced in lab curriculum (Donohue, 2014) among other changes to pedagogy and instruction.

Here, we address the challenges of teaching open source web mapping in higher education through a case study university course. Specifically, we report on our efforts to redesign and evaluate the course’s laboratory curriculum to leverage Open Web Platform technologies now widely in use for web design and web map production. To account for the increased range of competencies required for development on the Open Web Platform, we applied the constructivist pedagogies of scaffolding and spiralled curriculum, which build on prior understanding and progressively increase student independence and confidence. The revised lab curriculum engaged students in active learning and problem-solving to achieve four learning objectives related to open source web mapping: computational thinking, competence, confluence, and confidence. We evaluated the curriculum using a mixed methods approach that included an instructor observation log, direct feedback from students, and a detailed exit survey eliciting evidence of growth of student competency in web mapping.

In this paper, we demonstrate how constructivist instructional design can be adapted to teach technology skills that are increasingly in demand within the field of geography, in higher education generally, and in the broader information economy (Raja, 2014). We provide a unique (to our knowledge) in-depth case study of how the use of open source technologies as teaching tools can shape instructional design for cartography and geographic information science courses. We also provide resources for instructors of similar courses: we propose core learning objectives for courses that teach web mapping, highlight trouble spots for instructors to be aware of, and outline the sequence of laboratory topics with which we experienced success.

The following section describes the context of the lab curriculum and the pedagogical concepts applied in its redesign. In the third section, we describe the three-part curriculum evaluation procedure. We then discuss key findings regarding the curriculum’s benefits and stress points. We conclude with the revisions we have made to the lab curriculum based on the evaluation that will be implemented in the next offering of the course.

**Background: mapping on the open web**

Web maps are valuable and widespread media for presenting geographic information to the public. While web maps have been commonplace since the advent of the Internet several decades ago (Peterson, 2008), the underlying web mapping technology has undergone multiple, disruptive transitions during this time (Muehlenhaus, 2013). The first web maps were scanned static images published to the World Wide Web through hypertext documents. In contrast, web maps today include multiple geographic scales, adapt to the capabilities of the user’s device, change appearance in response to user interaction, and show real-time updates to their underlying data sources (Roth et al., 2014).

These dynamic, interactive web maps execute their programmatic instructions in one of two ways: via a third-party browser plug-in, or via instructions sent to the browser itself (Muehlenhaus, 2013). Third-party browser plug-ins include Java Virtual Machine, Microsoft Silverlight, and Adobe Flash Player. Web maps built to run in a plug-in are created using integrated development software packages such as Adobe Flash or Eclipse, which compile the code into a single executable file to be delivered over the internet. This type of web map
has the advantages of maintaining a consistent look and feel across browsers and operating systems, executing smoothly once loaded regardless of bandwidth and hardware capabilities, and only requiring the developer to be familiar with one authoring environment and a limited suite of development tools (Lienert, Jenny, Schnabel, & Hurni, 2012; Peterson, 2008).

The second type of web map uses the Open Web Platform, which consists of royalty-free technologies that directly enable the web, including Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), Scalable Vector Graphics (SVG), and JavaScript (Hu, 2008; World Wide Web Consortium [W3C], 2015). Open Web Platform maps are built using a text editor to code web page documents in HTML, style them using CSS code, render vector graphics using SVG code, and implement interactivity using JavaScript, the native scripting language of the Web, along with a wide range of extendable code libraries and frameworks. The code is delivered piecemeal over the internet and compiled and executed directly in the user's browser. Table 1 provides an overview of the range of tools and techniques used for mapping on the Open Web Platform, and thus the range of technical competencies that need to be instructed in web mapping curriculum in addition to mapping-specific topics.

Applications built on the Open Web Platform conform to the concept of free and open source software (FOSS), or programs and technologies (web or otherwise) that give users the freedom to run, copy, distribute, study, change, and improve them without notifying or paying royalties to prior distributors (Free Software Foundation, 2016). FOSS products may be monetized and distributed commercially, but the program source code must remain accessible and the software must be licensed for modification and redistribution to be considered free and open source (Gaff & Ploussios, 2012). FOSS follows an ethos of personal liberty and voluntary collaboration independent of the interests of business or state actors (Stallman, 2015).

Free and open source web mapping began in the mid-2000s with “map hacks” and mashups that demonstrated the versatility of Open Web Platform applications such as Google Maps and MapQuest (Gibson & Erle, 2006). This approach has since exploded into dozens of mapping tools built on the Open Web; Figure 1 shows several of the most used such tools (Crampton, 2010). Simultaneously, web mapping and web application development generally have shifted dramatically away from the use of third party browser plug-ins and toward sole reliance on the Open Web Platform (Muehlenhaus, 2013; Peterson, 2008, 2014; Tsou, 2011). Browser and operating system support for plug-ins began waning in 2010, when Apple announced that it would not support Flash Player on its mobile iOS platform (Jobs, 2010). With the explosive growth of mobile device usage, browser vendors sought to provide a consistent user experience across desktop and mobile platforms, leading them to phase out browser plug-ins and eventually discontinue support for them entirely (Smith, 2015).

FOSS technologies hold several advantages that have driven their uptake in commercial and government settings, and that currently make them attractive for instruction in geography, cartography, and GIS curricula. One obvious advantage is the low monetary cost compared to commercial software, which often can be unaffordable for students, especially once they graduate and are no longer covered by their school’s enterprise software licenses. However, FOSS technologies like the Open Web Platform hold pedagogical benefit in addition to cost savings. Teaching FOSS may better facilitate higher level learning objectives and group collaboration, as students can learn from other developers and each other through the sharing, reviewing, and manipulating of existing code (a process that also more closely mimics real-world development than prefabricated lab assignments). FOSS technologies
Table 1. Glossary of acronyms and technical terms for the Open Web Platform.

| Acronym/Term                  | Definition                                                                                                                                                                                                 |
|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Open Web Platform             | The collection of royalty-free technologies which enables the World Wide Web                                                                                                                              |
| Open source software          | Computer software with available source code that legally may be run, copied, studied, modified, and redistributed royalty-free to anyone for any purpose                                                        |
| HTML                          | Hypertext Markup Language, the standard computer language used to create the content of web pages                                                                                                         |
| CSS                           | Cascading Style Sheets, a computer language used to specify the presentation of content on a web page                                                                                                |
| JavaScript                    | A computer programming language used to specify the behavior of a web page                                                                                                                               |
| SVG                           | Scalable Vector Graphics, an image format for vector (linework) graphics that integrates with HTML and is commonly used on the Web                                                                              |
| Code library                  | A package of computer programming code that performs specific functions that are not otherwise readily available in the chosen programming language                                                                 |
| JSON                          | JavaScript Object Notation, a data file format based on JavaScript                                                                                                                                          |
| GeoJSON, TopoJSON             | Two variants of the JSON file format that are designed to encode geospatial data. The GeoJSON format uses non-topological lists of coordinates for its feature geometries, while TopoJSON encodes topological relationships between spatial features |
| XML                           | EXtensible Markup Language, a human-readable computer language used to encode information. HTML and SVG are variants of XML                                                                 |
| jQuery                        | A commonly used JavaScript code library that enhances a JavaScript program's access to web page content                                                                                                    |
| AJAX                          | Asynchronous JavaScript and XML, a JavaScript programming technique that enables two-way communication between the JavaScript program (executed by the user's browser) and a web server without requiring the browser to reload the web page. AJAX enables modern web mapping by allowing slippy map image tiles to be dynamically requested from a server as they are needed by the user |
| DOM                           | Document Object Model, a technical specification describing the content, structure, and properties of web page components as well as how to dynamically add, manipulate, and delete web page components  |
| GitHub                        | A commercial web service that combines the Git version control system with remote storage of software source files. GitHub has become a highly used platform for developers to share and collaborate on open source software applications |
| CSV                           | Comma-Separated Values, a data file format based on plain text                                                                                                                                               |
| KML                           | Keyhole Markup Language, a data file format based on XML that is used by the Google Earth software application                                                                                             |

Note: This table provides a reference to the range of tools and technologies used when developing maps on the Open Web Platform, and thus represents the list of technical competencies that must be instructed in web mapping curricula in addition to mapping-specific topics.

Figure 1. A selection of Open Web Platform tools and technologies that may be used to make web maps.
often have robust developer communities dedicated to maintaining and improving the software, and train students to conform to open standards set by the developer community (Gaff & Ploussios, 2012; Open Source Initiative, 2006).

However, certain disadvantages of FOSS technologies and the Open Web Platform have partially inhibited their uptake in geography, cartography, and GIS higher education settings. Mapping on the Open Web Platform utilizes a dizzying and ever-changing array of development environments, data formats, graphics specifications, scripting languages and libraries, and publishing platforms (Donohue, 2014). Accordingly, there is a greater learning curve in understanding and integrating this range of competencies than with highly robust commercial software packages. Learners often have few early successes to reinforce learning and build confidence (Roth et al., 2014). This is particularly true for students and professionals only trained in desktop GIS and graphic design software, as lack of experience with “thinking like a computer” acts as a major barrier to getting started with the programming required by the Open Web Platform. Finally, given the expert user focus of many FOSS communities, learning materials for FOSS technologies tend to be written at a higher level than those for proprietary programs and are thus less approachable for newcomers.

**Case study: an Open Web Platform cartography curriculum**

**What to teach: open web mapping technologies and learning objectives**

The case study lab curriculum discussed here is part of a four-credit course designed for senior undergraduate and graduate students who have had at least one prior course in both cartographic design and object-oriented programming. The course is divided into lecture and laboratory components, each allocated two credit hours. Lectures walk through tenets of user interface (UI) and user experience (UX) design, reviewing concepts and best practices largely drawn from other fields, such as human–computer interaction, information visualization, usability engineering, and web design (Roth, 2013). The lab sections serve as the technical complement to lectures, introducing the skills needed to implement design concepts and examples presented in lecture. The course syllabus and schedule are provided as supplementary materials for additional background.

In 2012, we conducted a study of a broad swath of then-emerging web mapping tools in order to determine which combination of these tools best supported the design concepts and examples presented in the course lecture (Roth et al., 2014). The results of this research led us to select the Leaflet (http://leafletjs.com/) and D3 (http://d3js.org) JavaScript code libraries as the primary tools for introducing web mapping, replacing Adobe Flash. These open source libraries are widely used for client-side (i.e., in the browser) web mapping at the time of this writing, but were nascent in 2012. Both libraries are open source and freely available on GitHub (http://github.com).

Our prior laboratory curriculum consisted of four multi-week assignments reflecting the topics presented in the lecture component of the course: (1) an “Animation Challenge” that introduced students to the Flash authoring environment and the use of ActionScript to create cartographic animation, (2) an “Interaction Challenge” that introduced user interface controls for sequencing through time intervals and retrieving thematic information from the map, (3) a “Coordinated Visualization Challenge” that taught how to coordinate user interactions across data views in a simple geovisualization application, and (4) an open-topic
final assignment that required students to collaborate in groups of three, applying what they
had learned to the development of an interactive map addressing a real-world problem.

We rewrote the instructions for the laboratory assignments over the Summer and Fall
terms of 2012 to make use of Leaflet and D3 instead of Flash. The first two assignments
were transitioned to a Leaflet code base, and eventually collapsed to a single, longer exercise
(Donohue, Sack, & Roth, 2013). The third assignment was recreated using D3 because of
its utility for creating both maps and non-map data visualizations, and to present students
with an opportunity to transfer learned skills to a new technology context (Sack, Donohue,
& Roth, 2014). The final project was only changed insofar as the expectation that students
would make use of the Open Web Platform with Leaflet and/or D3 instead of Flash. The
course was taught with the rewritten lab assignments in Spring of 2013. While the under-
lying technology changed from Flash to the Open Web Platform, we did not change the
design considerations of assignments or the sequence of topics introduced in the course.

Some students were quite successful in the 2013 course, but quite a few struggled to learn
and integrate the new Open Web Platform tools and technologies. Many students had dif-
ficulty extending the skills introduced in the lab assignments to generate custom solutions
for their independent final projects (Donohue, 2014). The sheer range of concepts that
students were expected to master caused frustration, resulting in learned hopelessness for
some. Learning D3 in particular turned out to be overwhelming for a number of students
given its more complex programmatic approach than Leaflet. By the end of that semester, it
was clear that we needed to shift our attention from what to teach in an open web mapping
curriculum, to how to teach the competencies associated with the Open Web Platform. We
identified the need for a more formally structured lab curriculum with greater attention to
key stumbling blocks for comprehension of this highly technical material.

First, we sought to identify where students struggled with threshold concepts, or key ideas
that must be grasped to advance to a new level of competency (Bampton, 2012; Fouberg,
2013). Threshold concepts include ideas that are isolated, difficult, foreign, or counterintu-
itive to students and thus barriers to advancing understanding of a subject (Perkins, 1999).
Because computer science is rarely taught in American primary or secondary schools, web
map development is riddled with threshold concepts for many students. This prevalence
of threshold concepts required careful rethinking of our instructional methods and lesson
sequence.

Second, we outlined four overall learning objectives that are necessary for mastery of
open source web mapping:

1. **Computational thinking**: understand the flow of execution in a computer program
   and solve problems in the code.
2. **Competence**: apply Open Web Platform mapping tools successfully across a range
   of mapping contexts.
3. **Confluence**: analyze how data, representation, and interaction tools integrate
   across the web mapping workflow.
4. **Confidence**: evaluate one's achievements positively and trust one's ability to
   improve at difficult web mapping tasks.

The use of Open Web Platform tools requires **computational thinking**, or the ability to
reason like a computer algorithm, visualize the necessary information processing tech-
niques, and decompose large computational tasks into a logical series of smaller steps
Once these steps have been identified, students must build competence in applying the tools of the Open Web Platform to complete them. Finding the right solutions often requires students to make use of multiple tools that accomplish different tasks, so students need to analyze how these tools work together to integrate them in an effective technological confluence. Finally, through learning, encouragement, and eventual success, students build confidence in their abilities. These processes are nonlinear and iterative; each new discovery of a solution builds computational thinking, competence, confluence, and confidence.

**How to teach it: scaffolded instruction and spiral curriculum**

In redesigning the lab curriculum to better conform to FOSS tools and concepts and to support the four identified learning objectives, we looked to constructivist educational theory, which emphasizes direct experience in authentic learning environments, conceptualization of new information as cognitive schemata, and metacognition, or the learner’s self-reflection of their learning process (Fouberg, 2013; Meece, 2002; Neisser, 1976). Constructivism suggests that an effective teaching environment for professional web development skills should facilitate direct experience with the tools, be relevant to students' interest domains, and allow for student control over the learning experience (Ellis, 2003). This approach to learning fits well with the ethos of FOSS, which emphasizes personal initiative, collaboration, and experimentation in software use and development.

We redesigned the lab topic sequence for the 2014 course offering to utilize the constructivist strategy of scaffolded instruction, a process by which an instructor provides calibrated assistance to students who are learning to master a difficult task or problem (Meece, 2002; Wood, Bruner, & Ross, 1976). This approach is based on the concept of the zone of proximal development, or the gap between students’ current competencies and their potential abilities given instructor and peer assistance (Vygotsky, 1978; Figure 2). Scaffolding seeks to meet students within this zone, introducing a task that students cannot accomplish on their own and challenging them to move toward a more mature understanding of the subject matter and eventual mastery (Palincsar, 1986). Initially, the instructor may demonstrate partial solutions to the problem and break the task down into highly structured and simplified steps that students can complete. Over time, lessons on the subject are progressively more generalized to require less structure and assistance—a withdrawal of the scaffold. When students have moved out of the zone of proximal development by adequately mastering a particular skill or topic, a new, more challenging topic may be introduced.

We integrated the scaffolding strategy with a spiral curriculum approach, which involves revisiting key concepts throughout the curriculum to reinforce them until learners achieve mastery (Foote, 2011; Reigeluth, 2007). Each spiral consists of a new experience or set of experiences accompanied by student reflection and synthesis, with each rotation of the spiral (or unit) involving a greater level of challenge. The new lab curriculum was broken into a series of short lessons that iteratively built on prior skills over the course of four larger course units: (1) a preliminary tutorial assignment and two weeks of lessons in preparation for the first major lab assignment, (2) lessons and work time pertaining to the Leaflet lab assignment, (3) lessons and work time pertaining to the D3 lab assignment, and (4) the final project work period (Table 2). Each of these divisions in the curriculum could be considered
a rotation of the larger spiral for the course, as each one involved introducing new topics that built on skills learned in an earlier unit.

For the early weeks of each unit, the lesson format included a mix of lecture-style direct instruction, coding demonstrations by the instructor, and weekly active learning exercises that students were required to complete. Students were provided with example code from each lesson and pointed to relevant online tutorials, examples, and documentation to complete the exercises and solve problems. As each unit progressed and students’ skills developed, direct instruction diminished and students spent more of the lab period learning and problem-solving the assignment independently with over-the-shoulder assistance, following the scaffolding model. Over the course of the curriculum as a whole, each unit involved progressively less time spent on direct instruction, with the final project weeks dedicated solely to individualized assistance with problem-solving as needed. The complete lab schedule is provided as a supplementary material.
Table 2. Outline of 2014 web mapping curriculum topics.

| Pre-lab                  | Lab 1 (Leaflet)                              | Lab 2 (D3)                                         | Final project                      |
|--------------------------|----------------------------------------------|---------------------------------------------------|-----------------------------------|
| **Week 0**               | **Week 3**                                   | **Week 6**                                        | **Weeks 10–13**                   |
| (no lab meeting)         | Using Reference Documentation               | GitHub Concepts and Web Hosting                   | Individualized Assistance         |
| Online JavaScript Tutorial| Online Forums and Examples                  | SVG Basic Elements and Attributes                 |                                   |
|                          | Slippy Map Tile Concepts                     | D3 API Reference and Examples                      |                                   |
|                          | Leaflet Basic Concepts and Methods           | D3 Core Selectors and Generator Functions         |                                   |
| **Week 1**               | **Week 4**                                   | **Week 7**                                        | **Week 14**                       |
| Text Editors             | Using Developer Tools for Debugging          | Final Project Group Selection                      | Final Project Completion           |
| Directory Structure      | Custom UI Elements and Interactions          | D3 Interactions                                    |                                   |
| HTML Basic Elements and Attributes | HTML Basic Elements and Attributes | D3 Geography                                      |                                   |
| CSS Basic Style Rules    | JavaScript Basic Concepts                    |                                                  |                                   |
| jQuery Basic Concepts    | jQuery Basic Concepts                        |                                                  |                                   |
| **Week 2**               | **Conference Week**                          | **Week 8**                                        |                                   |
| Data Levels and Types    | Conference Week (no lab meeting)             | Workshop Final Project Proposals                  | Individualized Assistance         |
| Geographic Coordinates   |                                              |                                                  |                                   |
| Data Language Specifications | Review: TopoJSON, D3 Projections, Debugging |                                                  |                                   |
| AJAX (Asynchronous JavaScript and XML) | Individualized Assistance (final week for Lab 1) |                                                  |                                   |
|                          | Individualized Assistance                    |                                                  |                                   |
|                          | (final week for Lab 1)                       |                                                  |                                   |

Note: The first column shows introductory topics presented prior to the first lab assignment, the second and third columns represent the time frame of each lab assignment, and the fourth column shows the time frame for the final project.
Curriculum evaluation

Methods

We conducted the lab curriculum evaluation with the 24 students who completed the course in Fall Semester, 2014. We used three evaluation instruments to collect a mix of qualitative and quantitative data regarding how students learned the material presented, allowing for triangulation between instruments to strengthen the reliability of our findings (Merriam, 1988). Each of these is described below.

Instructor observation log

Throughout the course, we maintained an observation log containing reflections from each lab period. In a constructivist setting, lessons are a means to an end rather than the end in themselves, and thus rarely are executed exactly as planned (Spady, 1994). A detailed observation log is useful for generating a record of how the curriculum was actually implemented in the classroom, identifying unintended consequences of the curriculum, and obtaining direct evidence of students’ mastery of the skills introduced (Lewy, 1977). The logs captured challenges encountered during lesson implementation, successful teaching experiences, and student reactions to the material. Critical incidents recorded in the logs provided insight into students’ learning patterns and where they encountered successes or difficulties.

Student feedback compositions

Each of the two major lab assignments gave students the option to submit for extra credit a minimum half-page composition describing their struggles, “aha!” moments, and suggestions for improvements to the assignment. We received 14 of these descriptions for each lab and conducted a simple content analysis, coding the content by four categories: (1) concepts described by students as difficult, (2) other problems, (3) “aha!” moments, and (4) other benefits perceived as coming from the assignments (Suchan & Brewer, 2000). Within these categories, we grouped similar statements into themes and recorded the frequency of each theme.

Exit survey

We made use of an online exit survey administered to students at the end of the course to confirm and interpret findings generated by the other instruments (Merriam, 1988; Suchan & Brewer, 2000). Twenty-three students completed the exit survey, out of 24 who completed the course. The survey included five different sections:

1. The first section asked students to rate their competence with 20 Open Web Platform mapping tools before and after taking the course. Fourteen of these were used as tools in the lab curriculum, and six were not, providing controls against which to measure the lab curriculum’s impact on competence (Learning Objective 2). To collect the competence ratings, we used a seven-point Likert scale comparable to the competency scale in Prager and Plewe (2009); our scale was a continuum from “I have never used this specification” (1) to “expert level knowledge” (7).

2. The second exit survey section asked students to rate how challenging they found each of the tools covered in the curriculum, also using a seven-point Likert scale.
This helped to indicate the level of computational thinking required by each tool (Learning Objective 1).

(3) The third section asked students to reorder the topics in any sequence that made more sense to them (see Table 2) and to suggest topics that needed more reinforcement or were unnecessary and could be removed. The topic sequence reordering was included to indicate what sequence would best facilitate confluence of the covered Open Web Platform tools (Learning Objective 3).

(4) The fourth section asked students to rate their reliance on various learning resources and the helpfulness of each of those resources.

(5) The fifth section narrowed in on each of the two major lab assignments and final project. For each assignment, we asked what parts of the assignment students found challenging, what resources they relied on for help, whether they felt they had learned the material, and their overall emotional experience (a seven-point Likert scale from “extremely negative” to “extremely positive”). We collected data on students’ emotional experiences to assess student morale for each assignment and overall growth in student confidence (Learning Objective 4).

Results

Instructor observation log
Here, we report the key insights from the log in chronological order (Figure 2), reflecting the progressive increase in students’ progress toward the learning objectives throughout the course. Several technical terms are necessary for this description; refer to Table 1 for a description of terms that may be unfamiliar.

During the Week 1 lessons, many students found it challenging to grasp the Document Object Model (DOM), the conceptual framework that describes the internal structure of website elements. Understanding the DOM requires the ability to visualize abstract data structures, a key component of computational thinking (Learning Objective 1; Raja, 2014). Students also found it difficult to understand when to use syntax belonging to JavaScript versus the syntax of jQuery and other code libraries. Identifying the correct syntax for a particular code library is necessary in order to apply the methods provided by library. In Week 2, students had difficulty grasping Asynchronous JavaScript and XML (AJAX). Using AJAX requires careful attention to the order of execution of tasks in a script, as some tasks will only execute after data is received from a server (i.e. asynchronously with the rest of the script). The difficulty encountered in teaching the DOM, syntax, and AJAX indicated a need to refine and expand those lessons, and to better integrate debugging and browser development tools into the initial unit.

The lessons for Weeks 3 and 4 largely relied on instructor demonstrations using the Leaflet library that aimed to quickly boost student competence and confidence in the use of Leaflet (Learning Objectives 2 and 4). Debugging using in-browser developer tools was introduced during Week 4 as a homework exercise. In retrospect, had they been introduced earlier, students could have used these tools immediately to help visualize and grasp the difficult concepts associated with the DOM, syntax use, and AJAX. Week 5, the final week students were given to complete the Leaflet lab assignment, primarily consisted of one-on-one assistance to students, representing a partial removal of the scaffold.
The first lesson of Week 6, the introduction to GitHub, proved to be challenging to teach and to learn. The GitHub version control system can be accessed through both a command-line application and a graphical application. As many students felt uneasy with command-line tools, we chose to introduce GitHub through the graphical interface; however, this application was difficult to set up on a network of lab computers, poorly documented by its creators, and not very forgiving of beginner mistakes. Additionally, GitHub uses a different workflow than other file-sharing software, in which software developers copy or “clone” a repository of source files, make changes to the repository on their own computer, then “merge” those changes with the original repository stored online. Some students experienced difficulty reconciling conflicting file versions when they occurred in the course of collaborative work on the final project.

In contrast to GitHub, the introductory D3 lessons presented in Week 6 were among the most successful teaching experiences reported in the log. The “D3 Core Selectors and Generator Functions” lesson was divided into separate conceptual chunks or “best practices” that were introduced clearly and sequentially and built on data concepts covered in prior lessons. This segmented approach allowed students to focus on manageable steps toward mastering the D3 library and integrating it with other Open Web Platform technologies, creating an effective scaffold to competence (Learning Objective 2) and confluence (Learning Objective 3) that were major hurdles in the 2013 offering of the course.

Over the following three weeks, it became evident that while students’ success with learning D3 overall exceeded our expectations, students did experience challenges with three specific components of the D3 lab assignment. First, a number of students had difficulty making use of the TopoJSON data format, mostly due to unanticipated crashing of the online file conversion software they were directed to use – a risk of using very new open source tools that are not yet well-established. Second, D3’s implementation of geographic projection parameters – covered in Week 7’s “D3 Geography” lesson – proved the most conceptually difficult aspect of D3 and required extra review in Week 9, indicating a need to further refine that lesson. Finally, a number of students experienced difficulty debugging script errors, further suggesting the need to integrate developer tool training early in the curriculum. However, student confidence (Learning Objective 4) seemed to soar during the D3 unit; as recorded in the instructor log, “[e]ven when [students] need [instructor] advice to get something working, the attitude generally seems to be I’m learning and know I’ll get beyond this rather than helplessness or giving up.”

**Student feedback compositions**

The 28 student feedback compositions that were turned in with the two lab assignments gave a glimpse into students’ perspectives on their learning processes. The results below are ordered by four categories of statements we identified – (1) difficult concepts, (2) non-conceptual problems, (3) “aha!” moments, and (4) other positive experiences (Table 3). Within each category, the most common statements for each lab assignment and those common to both lab assignments are discussed in turn.

Students reported 45 themes regarding difficult concepts, with 19 of these mentioned in two or more essays. The most frequently mentioned themes were difficulty extending the Leaflet lab product beyond the example code provided in the assignment tutorial \((n = 7)\) and attempting to add a Leaflet plug-in to accomplish a task not included in the provided example code. Both themes indicate the difficulty of developing confluence (Learning Objective 3)
early in the scaffolded and spiralled lesson sequence. Other Leaflet themes included understanding the basics of JavaScript, building a dynamic legend, finding creative ways to solve problems, and improving the efficiency of custom code (n = 2 for each). The D3 assignment compositions included fewer conceptual difficulties, with the most common (n = 4) using script to manipulate and combine data objects – a concept that is related to JavaScript and computational thinking (Learning Objective 1) but not to D3 itself. One student summed up a major challenge related to confluence (Learning Objective 3):

I didn’t fully understand when I was using a Javascript, Jquery or Leaflet function. I understand the difference between these things for the most part, but maintaining proper syntax throughout all of them is extremely difficult, especially when you are also including html and css.

Students reported 33 themes regarding other problems that were not necessarily conceptual in nature. By far, the most mentioned theme was the presence of unresolved issues with the map display at the time of lab submission (n = 8), although the display issues varied widely in their specific nature. Otherwise, the most common problem themes during the Leaflet assignment were the choice of a data-set that did not fit the assignment scenario, a feeling of frustration with the student’s own lack of understanding, and the feeling of not having enough practice with coding to complete the assignment successfully (n = 3 for each). Such problems could negatively impact student confidence (Learning Objective 4) if not addressed.

Students reported 24 themes related to “aha!” moments. The most common themes included discovering how to track the order of execution in the script, success using online plug-ins and examples, and learning how to test script effectively to solve problems in the code (n = 3 for each). Four students reported that working through errors independently, with a peer, or with the instructor generated “aha!” moments, showing the importance of active learning and collaboration. One student reflected on cementing a key computational thinking skill (Learning Objective 1): “I was getting overwhelmed by the enormity of trying to solve huge problems. I needed to break it down and solve things one at a time, not all at once.” The “aha!” moments in the Leaflet compositions reflected more conceptually basic realizations such as how to use a particular library method or perform a scripting task, whereas many of the D3 compositions reflected higher level creative problem-solving insights – a finding in line with the progressively empowering nature of the scaffolded curriculum.

We identified 26 themes regarding positive experiences that were not attached to a specific “aha!” moment. The most common theme was an appreciation for the clarity and helpfulness of the lab assignment tutorials (n = 5). Four students (two for each assignment) reported that they felt more confident using JavaScript than they had before starting the assignment (Learning Objective 4). Multiple students remarked that they liked and understood D3 better than Leaflet, which seems to run counter to the experience of the 2013 course, and points to the success of the prior lessons at building the scaffold necessary for understanding D3.

| Theme                      | Themes lab 1 | Freq. lab 1 | Themes lab 2 | Freq. lab 2 | Themes total | Freq. total |
|----------------------------|--------------|-------------|--------------|-------------|--------------|-------------|
| Difficult concepts         | 23           | 44          | 31           | 38          | 45           | 82          |
| Other problems             | 21           | 33          | 19           | 30          | 33           | 63          |
| Aha! moments               | 12           | 16          | 14           | 18          | 25           | 34          |
| Other positive experiences | 12           | 16          | 17           | 25          | 26           | 41          |

Table 3. Statement themes and frequencies from student feedback compositions.
Exit survey
The exit survey included sections on: (1) competence with Open Web Platform tools, (2) difficulty of learning the tools, (3) the sequence of lesson topics, (4) learning resources, and (5) lab assignments.

To analyze the first section of the exit survey, we compared the competence ratings for each of 20 Open Web Platform tools that students gave themselves at the beginning of the course to their ratings for the same tools at the end of the course using unpaired t-tests (Table 4). All but one of the tools that were covered by the scaffolded and spiralled curriculum exhibited a significant increase in mean student expertise at the 95% confidence level, mostly from low to moderate levels. The one covered tool that students did not significantly gain proficiency with, the CSV data format, most likely was familiar to students from prior GIS courses, as its initial mean expertise was moderate (4.6 out of 7). The lowest mean expertise as a final outcome in this group of tools was for AJAX (2.9), indicating that students still found this a difficult tool to use even after grappling with it extensively. In contrast, for the six tools that were not explicitly covered by the curriculum, there were only slight increases in mean expertise, and all final means fell below 3. Only one of these tools, Mapbox Studio/TileMill (the name changed with a new version released during the course), had an increase that was statistically significant ($p = 0.01$); this may have been due to some students choosing to use this tool independently to complete their final projects.

In the second survey section, assessing the difficulty of learning each tool, D3 was rated the most difficult tool to learn, with a mean difficulty rating of 5.2 out of 7. It also had the lowest standard deviation (1.17), indicating most students rated it highly difficult (Table 5). However, these difficulty ratings do not necessarily indicate how positively students viewed their learning experiences with each tool. When asked to rate their overall emotional experience with the lab assignments and final project in the fifth survey section, the mean positivity rating rose for each successive assignment. The mean positivity rating for the D3 lab assignment was higher than that of the Leaflet assignment (5.2 vs. 4.8), and all students rated their experience with the D3 assignment a 3 out of 7 or better (Figure 3). These results indicate that students found D3 challenging yet rewarding, and echo the progressive improvement of confidence recorded in the instructor log (Learning Objective 4).

In the third survey section, students were asked to reorder the curriculum topics in a way that made most sense to them and to comment on which topics needed more reinforcement or were not useful. The mean positions indicated that “using in-browser developer tools to debug” and “basics of GitHub” should come earlier in the sequence (Table 6). This result reinforces the idea that these topics were threshold concepts foundational to subsequent course material. Several students found the “coordinate systems” ($n = 6$) and “data levels and types” ($n = 5$) lessons unhelpful, largely due to their redundancy with prior Cartography and GIS courses, indicating that these topics can be removed in the future to make room for greater coverage of identified threshold concepts.

In the fourth survey section, covering learning resources, several students ($n = 6$) indicated that instructor-led coding demonstrations – in which the instructor worked through a coding problem with students observing and following along – were not useful or were difficult to follow. This important observation was missed by the instructor logs and student essays, and suggests that coding demonstrations should be replaced with more experiential, student-controlled activities and greater emphasis on how to use online resources.
Table 4. Student-reported increase in expertise with Open Web Platform mapping tools.

| Open Web Platform tool | Mean/SD expertise prior to course | Mean/SD expertise after taking course | Covered by lesson topics? | Increase significant at 95% level? |
|------------------------|----------------------------------|--------------------------------------|---------------------------|----------------------------------|
| HTML                   | 3.26/1.76                        | 4.61/1.27                           | Yes                       | Yes                              |
| CSS                    | 2.87/1.52                        | 4.30/1.33                           | Yes                       | Yes                              |
| JavaScript             | 2.30/1.36                        | 4.04/1.33                           | Yes                       | Yes                              |
| jQuery                 | 1.83/1.27                        | 3.43/1.24                           | Yes                       | Yes                              |
| AJAX                   | 1.52/0.99                        | 2.87/1.18                           | Yes                       | Yes                              |
| The DOM                | 1.74/1.21                        | 3.30/1.18                           | Yes                       | Yes                              |
| Git/GitHub             | 1.74/1.36                        | 3.61/1.64                           | Yes                       | Yes                              |
| Leaflet                | 1.65/1.27                        | 4.17/1.11                           | Yes                       | Yes                              |
| D3                     | 1.17/0.65                        | 3.78/1.24                           | Yes                       | Yes                              |
| Google Maps API        | 1.78/1.00                        | 2.26/1.29                           | No                        | No                               |
| OpenLayers             | 1.35/0.93                        | 1.87/1.32                           | No                        | No                               |
| ArcGIS Online          | 1.61/1.27                        | 2.04/1.49                           | No                        | No                               |
| MapBox Studio/TileMill | 1.96/1.33                        | 2.96/1.22                           | No                        | Yes                              |
| CartoDB                | 1.43/0.95                        | 1.91/1.35                           | No                        | No                               |
| JSON                   | 2.17/1.53                        | 4.48/1.27                           | Yes                       | Yes                              |
| GeoJSON                | 1.91/1.53                        | 4.43/1.31                           | Yes                       | Yes                              |
| TopoJSON               | 1.61/1.41                        | 4.39/1.31                           | Yes                       | Yes                              |
| SVG                    | 2.04/1.52                        | 3.91/1.50                           | Yes                       | Yes                              |
| CSV                    | 4.57/1.56                        | 5.17/1.40                           | Yes                       | No                               |
| KML                    | 2.57/1.50                        | 2.96/1.69                           | No                        | No                               |

Table 5. Students’ mean difficulty ratings for Open Web Platform mapping tools, from most to least difficult.

| Open Web Platform tool | Mean difficulty | SD  |
|------------------------|-----------------|-----|
| D3                     | 5.22            | 1.17|
| JavaScript             | 4.52            | 1.83|
| AJAX                   | 4.43            | 1.41|
| jQuery                 | 4.13            | 1.49|
| Leaflet                | 4.13            | 1.32|
| The DOM                | 3.74            | 1.86|
| Git/GitHub             | 3.74            | 1.54|
| CSS                    | 3.48            | 1.68|
| SVG                    | 3.17            | 1.44|
| TopoJSON               | 2.83            | 1.44|
| HTML                   | 2.78            | 1.57|
| GeoJSON                | 2.70            | 1.49|
| JSON                   | 2.48            | 1.44|

Figure 3. Students’ overall emotional experiences while completing each lab assignment.
In the fifth section, covering the lab assignments and final project, 21 out of 23 students (91%) somewhat to strongly agreed that they “knew how to make an effective and well-designed web map using Leaflet” after completing the first lab (mean agreement of 5.2/7). 16 students (69%) somewhat to strongly agreed with the same statement for D3 (mean agreement of 4.8/7), which is in line with the perception that D3 was the harder of the two tools to learn. All but one of the 24 students who completed the course produced a portfolio-quality web map for the final project (Figure 4). Four students said they wanted more time to complete the final project, and five indicated that they had been challenged by various aspects of collaboration with their peers.

| Topic                                           | Mean/ St. Dev. | Topic order as taught (circles) vs. student suggestions (boxplots) |
|-------------------------------------------------|----------------|---------------------------------------------------------------------|
| 1. Local website directory setup                | 3.7/4.3        |                                                                    |
| 2. Basics of HTML                               | 3.8/4.0        |                                                                    |
| 3. Basics of CSS                                | 4.8/3.4        |                                                                    |
| 4. Basics of JavaScript                         | 4.7/3.0        |                                                                    |
| 5. Basics of jQuery                             | 7.8/4.2        |                                                                    |
| 6. Data levels and types                        | 8.7/4.9        |                                                                    |
| 7. Geographic coordinate systems                | 10.5/5.1       |                                                                    |
| 8. Data format specifications                   | 8.3/3.5        |                                                                    |
| 9. Asynchronous JavaScript and XML (AJAX)       | 10.5/2.8       |                                                                    |
| 10. Using online reference documentation        | 8.7/3.6        |                                                                    |
| 11. Using search engines and online resources   | 10.4/3.7       |                                                                    |
| 12. Slippy map tile concepts                    | 12.6/1.5       |                                                                    |
| 13. Basics of Leaflet                           | 12.6/2.9       |                                                                    |
| 14. Using in-browser developer tools to debug   | 9.6/4.7        |                                                                    |
| 15. Leaflet custom UI elements and interactions | 14.7/1.7       |                                                                    |
| 16. Basics of GitHub                            | 10.6/6.2       |                                                                    |
| 17. SVG elements and attributes                 | 15.7/3.0       |                                                                    |
| 18. D3 selections                               | 16.7/3.7       |                                                                    |
| 19. D3 generator functions and scales           | 17.8/3.9       |                                                                    |
| 20. D3 projections and path generators          | 18.3/4.0       |                                                                    |
Discussion

The curriculum evaluation measured the effectiveness of the scaffolded and spiralled lab curriculum in meeting the learning objectives, identified threshold concepts in need of reinforcement, and highlighted where shifts in the topic sequence could enhance the scaffolding. This section expands on our findings and how they have informed revisions to the curriculum.

Course objectives

Computational thinking

The qualitative data in the instructor observation log and student feedback compositions pointed to improved computational thinking skills among students, especially by the time of the second lab assignment's completion. Students described learning to use proper code syntax, thinking through the order of execution, breaking large problems down into smaller pieces to debug, and finding creative ways of problem-solving. Computational thinking on the D3 lab assignment in particular was improved by the new curriculum, although problem-solving abilities could have been enhanced by an earlier and more thorough lesson on debugging.

Competence

The instructor log and student feedback compositions demonstrated a noticeable improvement in student competence over the length of the course, which was confirmed by students’ competence ratings in the exit survey. Student competence moved upward from a low to moderate level; students did not judge themselves to be experts in any of the covered tools by the end of the course, which falls in line with what can be reasonably expected of a semester-long university course covering a difficult technical subject matter (Prager & Plewe, 2009).
Confluence
The student feedback compositions and open-ended questions in the exit survey suggested an expanded grasp of the confluence of Open Web Platform tools. A number of students discussed their processes of learning to manipulate data using script, understand and incorporate example code, recognize methods belonging to different code libraries, and use those libraries in concert with one another and with JavaScript, CSS, and HTML. Each lab assignment required students to integrate multiple Open Web Platform tools, including JavaScript, AJAX, and multiple code libraries (jQuery, Leaflet, D3, Queue, etc.). Final projects gave students a chance to choose which tools to integrate, and the results demonstrated students’ abilities to make use of multiple tools to produce a high-quality deliverable.

Confidence
The instructor log, student feedback compositions, and emotional experience ratings in the exit survey triangulate to show the growth of confidence in using Open Web Platform mapping tools. Students felt more positive about their work as the semester went on and they gained skills and confidence, in contrast to the dip in morale observed in the 2013 course offering. Given the high level of frustration and difficulty with D3 experienced by students in the 2013 course, the enthusiasm of the 2014 students in learning and using D3 confirmed the efficacy of our scaffolded and spiralled approach, which involved breaking the concepts down into manageable pieces and drawing on foundational material from prior lessons to build understanding.

Threshold concepts for mapping on the Open Web Platform
The curriculum evaluation identified several threshold concepts associated with mapping on the Open Web Platform that were difficult for many students in the course. These included:

- basic JavaScript syntax;
- the relationship between native JavaScript and code libraries, especially jQuery;
- the DOM;
- AJAX;
- integrating example code into custom code;
- D3 projection parameters;
- TopoJSON;
- debugging; and
- GitHub.

The first four threshold concepts are computational thinking skills that pose a challenge to new learners: becoming accustomed to the strictness of programming syntax, recognizing where to apply different methods, visualizing an abstract set of computational objects, and following the flow of execution in a program, respectively. Debugging requires all of these skills. D3 projection parameters, TopoJSON, and GitHub are specific tools that require the building of competence through becoming familiar with particular workflows. Integrating example code into custom code is a key skill associated with confluence; it requires understanding what parts of the example code are relevant to the task being attempted. Having highlighted these topics as threshold concepts will allow us to more carefully structure lessons, tutorials, and learning exercises around them in future iterations of the course.
The instructor log and exit survey revealed that two of these threshold concepts in particular—GitHub and debugging—should be reordered to earlier in the topic sequence so that they receive reinforcement through the higher level concepts built on top of them. Debugging is a necessary step in getting any computer program to run correctly, but was not introduced until the fourth week of the curriculum, well into the work period for the first lab assignment (Table 2). If introduced earlier and in a more structured manner, it may have saved some students problem-solving headaches and would have been reinforced throughout completion of the Leaflet lab assignment. Similarly, GitHub is a useful version control, collaboration, and code publishing system that would have received greater reinforcement had it been introduced early on and used throughout.

Our experience demonstrates that students can experience great success using FOSS for web mapping. However, the challenges attendant with integrating multiple open source tools in a code-heavy environment necessitate a strong emphasis on proper sequencing and scaffolding to overcome difficult threshold concepts. From early on, students need to be taught to use developer tools that can help them understand the development workflow and grasp difficult coding concepts. Such concepts, once introduced, need reinforcement through a spiralled approach that revisits prior concepts when teaching new ones. Our curriculum design approach met with success, but our evaluation showed us where improvements could be made. These changes are outlined below.

**Curriculum revisions**

Based on this improved understanding of threshold concepts, we have revised the curriculum for future iterations of teaching. Table 7 shows the revised topic order and highlights lessons that have been expanded upon and/or moved. The “Basics of HTML” and “Basics of CSS” lessons have been removed due to their inclusion in another cartographic design course and the availability of remedial online training. Per student feedback, “Data levels and types” and “Geographic coordinate systems” have likewise been removed as redundant with other courses. The “Basics of GitHub” lesson has been moved to the very beginning of the course, and GitHub will be utilized as a platform for submission of student work throughout the course. However, the “Web Hosting” part of that lesson has been separated and moved to the end of the topic sequence. “Using Developer Tools for Debugging” has also been moved up significantly in the sequence.

Several new topics represent subdivisions of prior topics meant to break down threshold concepts into more readily learnable chunks. A new “Exploring the DOM” lesson will be taught along with JavaScript and jQuery basics. The “AJAX (Asynchronous JavaScript and XML)” lesson has been separated into two lessons dealing first with AJAX concepts, then with its application. “Using Reference Documentation” has been given more structure and refocused to take advantage of the easy-to-use Leaflet documentation and tutorials, while the “Online Forums and Examples” lesson has been broken into separate lessons on examples and forums. Finally, three topics that needed extra review later in the course—TopoJSON, D3 projections, and debugging—have been reinforced in the revised curriculum, with TopoJSON included in a new lesson on “D3 Helpers” and the latter two topics given their own dedicated lessons. Though not shown in Table 7, the Final Project will continue to be included in the residency version of the course, but due to the difficulty of group-based
Table 7. The revised 2016–2017 lesson sequence.

| Unit 1 | Unit 2 | Unit 3 |
|--------|--------|--------|
| **Module 1: Setting Up Your Workspace**<br>Boilerplates and Frameworks<br>Web Directory Setup<br>GitHub Setup (moved from Week 6)<br>Assigned: JavaScript Online Tutorial | **Module 4: Using Online Resources**<br>Leaflet Tutorials and API<br>Using Online Examples<br>Using Help Forums<br>Finding Tiles sets and Data | **Module 7: D3 Foundations**<br>D3 Selections and Blocks<br>Data<br>Scales, Axes, Text |
| **Module 2: Scripting and Debugging**<br>Exploring the DOM<br>JavaScript Basics<br>jQuery Basics<br>Debugging in the Developer Console (moved from Weeks 4 and 9) | **Module 5: Leaflet Interactions**<br>Making Leaflet Layers Dynamic<br>Zoom, Pan, and Retrieve Interactions<br>Sequence Interaction<br>Additional Interaction Operators | **Module 8: Mapping in D3**<br>D3 Helpers: TopoJSON, MapShaper & Queue<br>D3 Projections and Path Generators |
| **Module 3: Data and AJAX**<br>CSV, XML, and JSON formats and their geographic variants<br>AJAX Concepts and Syntax<br>AJAX Callback Functions | **Module 6: The Internal Logic of Leaflet**<br>Object-oriented JavaScript<br>Extending Leaflet Objects<br>Using SVG Graphics | **Module 9: Coordinated Visualizations**<br>Dynamic Map Styling<br>Drawing a Coordinated Visualization |
| **Module 10: Coordinated Interactions**<br>Dynamic Attribute Selection<br>Transitions<br>Linking Interactions Between Map and Chart<br>Deploying Your Geovisualization (moved from Week 6) | Notes: Lessons shown in italics were significantly expanded to better address threshold concepts identified by the curriculum evaluation. Underlined lessons were moved to a different point in the topic sequence due to student feedback.
student collaboration in distance learning, will not be part of the online version of the course (Bose, 2014).

Conclusion

The Open Web Platform has transformed not just the narrow field of web mapping, but the processes of research, knowledge creation, information dissemination, and even artistic expression writ large (Davidson, 2008). Whether or not geography students will ultimately seek jobs in the information economy, they increasingly need to be fluent in the skills necessary to utilize open source software and the Open Web, as these skills are increasingly in demand across academic and non-academic fields. Web mapping seems an obvious and practical entry point for budding Geographers to acculturate themselves to these highly technical toolsets. Well-established constructivist education principles such as scaffolding and spiral curriculum can provide an effective path to mastery of these toolsets through detailed attention to lesson structure and sequence as well as the use of authentic learning contexts.

The technical tools we teach must both be vehicles for correct application of design concepts and track rapidly changing industry standards and best practices (Ellis, 2003). For web mapping, this has meant a shift to open source software and integrated workflows that are both more flexible for creative exploration and more complex to learn and apply than the proprietary, all-in-one design software around which institutional knowledge has been built up. In the face of such rapid paradigm shifts, our curriculum development practices must be nimble as well as reflexive. Our evaluation process provides a framework for ensuring the curriculum’s continued effectiveness as we continue to adapt it to ongoing technology trends.

Acknowledgments

The authors wish to thank Richard Donohue for his integral role in the research study outlining what to teach and the Spring 2013 course offering.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

A portion of this research was funded by U.S. National Science Foundation Grant #155267 and the Wisconsin Alumni Research Foundation.

References

Bampton, M. (2012). Addressing misconceptions, threshold concepts, and troublesome knowledge in GIScience education. In D. J. Unwin, K. E. Foote, N. J. Tate, & D. DiBiase (Eds.), Teaching geographic information science and technology in higher education (pp. 117–132). West Sussex, UK: Wiley-Blackwell.

Bose, P. S. (2014). Technofetishism and online education: Globalizing geography through virtual worlds. Journal of Geography in Higher Education, 38, 28–39. doi:10.1080/03098265.2013.801070

Crampton, J. W. (2010). Mapping: A critical introduction to cartography and GIS. Malden, MA: Wiley-Blackwell.
Davidson, C. N. (2008). Humanities 2.0: Promise, perils, predictions. PMLA, 123, 707–717.
Donohue, R. G. (2014). Web cartography with web standards: Teaching, learning, and using open source web mapping technologies (Dissertation). University of Wisconsin-Madison, Madison.
Donohue, R. G., Sack, C. M., & Roth, R. E. (2013). Time series proportional symbol maps with Leaflet and jQuery. Cartographic Perspectives, 76, 43–66. doi:10.14714/CP76.1248
Ellis, H. J. C. (2003). Transfer of knowledge in a web design and development course. IEEE, 3, S1C-15–S1C-20. doi:10.1109/FIE.2003.1265911
Foote, K. E. (2011). Scope and sequence in GIS&T education: Learning theory, learning cycles and spiral curricula. In D. J. Unwin, K. E. Foote, N. J. Tate, & D. DiBiase (Eds.), Teaching geographic information science and technology in higher education (pp. 81–95). West Sussex, UK: Wiley-Blackwell.
Fouberg, E. H. (2013). “The world is no longer flat to me”: Student perceptions of threshold concepts in world regional geography. Journal of Geography in Higher Education, 37, 65–75. doi:10.1080/03098265.2012.654467
Free Software Foundation. (2016, January 1). What is free software? GNU Operating System. Retrieved from http://www.gnu.org/philosophy/free-sw.html
Gaff, B. M., & Ploussios, G. J. (2012). Open source software. Computer, 45, 9–11. doi:10.1109/MC.2012.213
Gibson, R., & Erle, S. (2006). Google Maps hacks (1st ed.). Beijing: O’Reilly.
Hu, S. (2008). Advancement of web standards and techniques for developing hypermedia maps on the internet. In M. P. Peterson (Ed.), International perspectives on maps and the internet (pp. 115–124). Berlin: Springer.
Jesdanun, A. (2015, December 7). Apple Maps, once a laughistock, now dominates iPhones. New York, NY: Associated Press. Retrieved from http://bigstory.ap.org/article/df90458e58564f19b4b7c8510f9baa67/apple-maps-once-laughingstock-now-dominates-iphones
Jobs, S. (2010, April). Thoughts on flash. Retrieved from http://www.apple.com/hotnews/thoughts-on-flash/
Lewy, A. (Ed.). (1977). Handbook of curriculum evaluation. Paris/New York: Unesco/Longman.
Lienert, C., Jenny, B., Schnabel, O., & Hurni, L. (2012). Current trends in vector-based internet mapping: A technical review. In M. P. Peterson (Ed.), Online Maps with APIs and web services (pp. 23–36). Berlin: Springer.
Meece, J. L. (2002). Child and adolescent development for educators (2nd ed.). Boston, MA: McGraw-Hill.
Merriam, S. B. (1988). Case study research in education: A qualitative approach (1st ed.). San Francisco, CA: Jossey-Bass.
Muehlenhaus, I. (2013). Web cartography: Map design for interactive and mobile devices. Boca Raton, FL: CRC Press.
Neisser, U. (1976). Cognition and reality: Principles and implications of cognitive psychology. San Francisco, CA: Freeman.
Open Source Initiative. (2006, September 19). Open standards requirements for software – Rationale. Open Source Initiative. Retrieved from https://opensource.org/osr-rationale
Palincsar, A. S. (1986). The role of dialogue in providing scaffolded instruction. Educational Psychologist, 21, 73–98.
Perkins, D. (1999). The many faces of constructivism. Educational Leadership, 57, 6–11.
Peterson, M. P. (Ed.). (2008). International perspectives on maps and the internet. Berlin: Springer.
Peterson, M. P. (2014). Mapping in the cloud. New York: The Guilford Press.
Prager, S. D., & Plewe, B. (2009). Assessment and evaluation of GIScience curriculum using the geographic information science and technology body of knowledge. Journal of Geography in Higher Education, 33, S46–S69. doi:10.1080/03098260903034012
Raja, T. (2014, August). Is coding the new literacy? Why America’s schools need to train a generation of hackers. Mother Jones, 39, 28–35.
Reigeluth, C. M. (2007). Order, first step to mastery: An introduction to sequencing in instructional design. In F. E. Ritter, J. Nerb, & E. Lehtinen (Eds.), In order to learn (pp. 19–40). Oxford, UK: Oxford University Press.
Roth, R. E. (2013). Interactive maps: What we know and what we need to know. *Journal of Spatial Information Science*, 59–115. doi:10.5311/JOSIS.2013.6.105

Roth, R. E., Donohue, R. G., Sack, C. M., Wallace, T. R., & Buckingham, T. M. A. (2014). A process for keeping pace with evolving web mapping technologies. *Cartographic Perspectives*, 25–52. doi:10.14714/CP78.1273

Sack, C. M., Donohue, R. G., & Roth, R. E. (2014). Interactive and multivariate Choropleth Maps with D3. *Cartographic Perspectives*, 57–76. doi:10.14714/CP78.1278

Smith, D. (2015, October 8). NPAPI Plugin Perspectives and the Oracle JRE. Retrieved from https://blogs.oracle.com/java-platform-group/entry/npapi_plugin_perspectives_and_the

Spady, W. G. (1994). *Outcome-based education: Critical issues and answers*. Arlington, VA: American Association of School Administrators.

Stallman, R. (2015). *Free software is even more important now*. GNU Operating System. Retrieved from https://www.gnu.org/philosophy/free-software-even-more-important.html

Suchan, T. A., & Brewer, C. A. (2000). *Qualitative methods for research on mapmaking and map use*, 52, 145–154.

Tsou, M.-H. (2011). Revisiting web cartography in the United States: The rise of user-centered design. *Cartography and Geographic Information Science, 38*, 250–257. doi:10.1559/15230406382250

Vygotsky, L. S. (1978). Interaction between learning and development. In M. Cole, V. John-Steiner, S. Scribner, & E. Souberman (Eds.), *Mind in society: The development of higher psychological processes* (pp. 79–91). Cambridge, MA: Harvard University Press.

Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry, 17*, 89–100.

World Wide Web Consortium. (2015, May 30). *Open Web Platform*. Author. Retrieved from https://www.w3.org/wiki/Open_Web_Platform